(19) JAPANESE PATENT OFFICE (JP)
(12) Japanese Unexamined Patent Publication (A)
(11) Patent Application (Kohyo) No.
H8-504284
(43) Publication Date: May 7 1996

(51) Int. Cl.[6]:	ID Code:	Internal Reference Nos.:	FI	
G06F 17/60				
G07F 7/12				
		9069-5L	G06F 15/21	340 Z
		0380-3E	G07F 7/08	C

Request for Examination: Not requested
Request for Preliminary Examination: Requested
(Total of 81 pages [in the Japanese text])

(21) Patent Application No.: H6-507504
(86) (22) Filing Date: 07/09/1993
(85) Translation Submission Date: 07/03/1995
(86) International Application No.: PCT/US93/08400
(87) International Publication No.: WO94/06103
(87) International Publication Date: 17/03/1994
(31) Priority Rights Claim No.: 07/941, 971
(32) Priority Date: 08/09/1992
(33) Priority Country: United States (US)

(71) Applicant: HNC Software Incorporated
5930 Cornerstone Court West
San Diego, California, USA 92121-3728
(72) Inventor: Krishna M. Gopinathan
9924 No. 2416 Kika Court
San Diego, California, USA 92129
(72) Inventor: Louis S. Biafore
5075 La Quinta Drive
San Diego, California, USA 92129
(72) Inventor: William M. Ferguson
4215 Feather Avenue
San Diego, California, USA 92117
(72) Inventor: Michael A. Lazarus
5030 No. 57 Lotus Street
San Diego, California, USA 92107
(72) Inventor: Anu K. Pathria
5445 Locksley Avenue
Oakland, California, USA 94618
(72) Inventor: Allen Jost
18605 Lancashire Way
San Diego, California, USA 92128
(74) Agent: Patent Attorney, [illegible] Koya
(and 2 others)
(81) Designated States: EP (AT, BE, CH, DE, DK, ES, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OA (BF, BJ, CF,

CG, CI, CM, GA, GN, ML, MR, NE, SN, TD, TG), AT, AU, BB, BG, BR, BY, CA, CH, CZ, DE, DK, ES, FI, GB, HU, JP, KP, KR, KZ, LK, LU, LV, MG, MN, MW, NL, NO, NZ, PL, PT, RO, RU, SD, SE, SK, UA, VN.

---

(54) [Title of the Invention] Fraud detection using predictive modeling

(57) [Abstract] An automated system and method for detecting transaction fraud using a predictive model such as a neural network to estimate individual customer accounts and identify likelihood of transaction fraud on the basis of relationships learned from among known variables. The system also outputs reason codes containing the contributions of the various variables related to a particular result. The system periodically monitors its performance and redevelops the model when its performance drops below a predetermined level.

1401 START

1402 STORING 1-DAY OF TRANSACTIONS

1403 OBTAINING CURRENT TRANSACTION DATA

1404 OBTAINING PAST TRANSACTION DATA, CUSTOMER DATA AND PROFILE DATA

1405 APPLICATION OF DATA TO NEURAL NETWORK

1406 OBTAINING FRAUD SCORE FROM NEURAL NETWORK

1407 FRAUD SCORE > THRESHOLD?

1405 ACCOUNT FLAGGED

1409 END

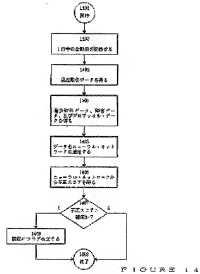


FIGURE 14

CLAIMS

What is claimed is:

1. A computer-executed process for detecting transaction fraud on a customer account, comprising the steps of:

developing a predictive model from past transaction data;

storing said predictive model in a computer-related medium;

obtaining current transaction data;

obtaining customer data; and

generating a signal indicating the likelihood of fraud in response to application of said current transaction data and said customer data to the stored predictive model.

2. The computer-executed process as claimed in claim 1, wherein said step of obtaining customer data comprises a step of accessing a database containing general customer data and a database containing customer transaction pattern data.

3. The computer-executed process as claimed in claim 1, wherein said step of obtaining customer data comprises a step of accessing just one profile database record containing customer transaction pattern data.

4. The computer-executed process as claimed in claim 3, wherein said profile database record also contains general customer data.

5. The computer-executed process as claimed in claim

1 in which said current transaction data and said customer data each comprise a plurality of elements, further comprising the steps of, for each of said current transaction data and said customer data:

assessing a contribution of elements related to a determined likelihood of fraud;

assessing a reason code value from the contributions related to the thus determined likelihood of fraud; and

generating a signal indicating said reason code value.

6. The computer-executed process as claimed in claim 1, further comprising the steps of:

comparing the determined likelihood of fraud with a current threshold; and

transmitting a fraud signal in response to the likelihood of said fraud exceeding said current threshold.

7. The computer-executed process as claimed in claim 1, further comprising the iterative steps of:

comparing said determined likelihood of fraud with a cascade threshold; and

generating another signal indicating the likelihood of fraud in response to application of said current transaction data and said customer data to another stored predictive model in response to the likelihood of said fraud exceeding said cascade threshold.

8. The computer-executed process as claimed in claim

1, further comprising the steps of:

monitoring a performance metric of the predictive model;

comparing said performance metric with a predetermined performance level; and

developing and storing a new predictive model from past transaction data in response to said predetermined performance level exceeding said performance metric.

9. The computer-executed process as claimed in claim 8, wherein said performance metric comprises:

a fraud detection rate measurement; and

a false response rate measurement.

10. The computer-executed process as claimed in claim 1, wherein said predictive model is a neural network.

11. The computer-executed process as claimed in claim 1, wherein said step of developing the predictive model comprises the substeps of:

obtaining past transaction data;

pre-processing said past transaction data and deducing past fraud-related variables; and

training said predictive model with said deduced past fraud-related variables.

12. The computer-executed process as claimed in claim 11, wherein said substep of training the predictive model comprises the iterative substeps of:

applying input data to said model;

ordering output data produced in response to a measurement of quality; and

adjusting an operation of said model in response to a result of said ordering step.

13. The computer-executed process as claimed in claim 12 in which said predictive model comprises a neural network having a plurality of intercoupled processing elements, each processing element comprising:

a plurality of inputs;

a plurality of weights related to corresponding inputs for generating weighted inputs;

means for combining said weighted inputs; and

a transfer function for processing said combined weighted inputs to generate an output.

14. The computer-executed process as claimed in claim 13 in which said substep of adjusting operation of the model comprises the substeps of:

selecting a partial subset of the weights to be decayed; and

decaying said selected partial subset of weights.

15. The computer-executed process as claimed in claim 14, wherein said substep of selecting a partial subset of the weights to be decayed comprises a step for applying and minimizing a cost function containing an interlayer gain multiplier that varies decay rate in response to the location of a weight within said network.

16. The computer-executed process as claimed in claim

15, wherein said cost function is represented by the following expression:

$$\frac{1}{2}\sum_{k\in D}(target_k - output_k)^2 + gl\sum_{i\in w}(c_i w_i^2 - \frac{1}{1+|w_i|})$$

here:

D expresses a data set;

$target_k$ expresses a target value for an element k of the data set;

$output_k$ expresses a network output for the element k of the data set;

g expresses the interlayer gain multiplier;

l expresses a relative importance of a complexity term;

w expresses a weight set;

w_i expresses a value of a weight i; and

c expresses a constant.

17. A computer-executed process for detecting transaction fraud on a customer account, comprising the steps of:

obtaining past transaction data;

pre-processing said past transaction data to deduce past fraud-related variables;

training a predictive model using said deduced past fraud-related variables;

storing said predictive model in a computer-related medium;

obtaining current transaction data;

pre-processing said current transaction data and deducing current fraud-related variables;

obtaining customer data;

pre-processing said customer data and deducing customer fraud-related variables; and

generating a signal in response to likelihood of fraud in response to application of said current fraud-related variables and said customer fraud-related variables to the stored said predictive model.

18. The computer-executed process as claimed in claim 17 in which said past fraud-related variables and said current fraud-related variables each comprise at least:

a factor obtained from data that refers to transaction dollar amounts pertaining to a fraud;

a factor obtained from data that refers to transaction dates and times pertaining to a fraud;

a factor obtained from data that refers to transaction approve and decline pertaining to a fraud; and

a factor obtained from data that refers to a risk group pertaining to a fraud.

19. The computer-executed process as claimed in claim 17, wherein said past fraud-related variables and said current fraud-related variables comprise at least:

a factor obtained from data that refers to a customer pertaining to a fraud; and

a factor obtained from data that refers to a merchant pertaining to a fraud.

20. The computer-executed process as claimed in claim 17, further comprising the steps of, for each subset of deduced said current fraud-related variables and deduced said customer fraud-related variables:

assessing a contribution of variables related to the determined likelihood of fraud to generate a reason code value;

assessing a reason code value from contributions related to each thus determined likelihood of fraud; and

generating a signal indicating said reason code value.

21. The computer-executed process as claimed in claim 17, wherein the predictive model is a neural network.

22. A computer-executed process for training a predictive model stored in a computer-related medium for predicting results on the basis of selected characteristics comprising the iterative steps of:

applying input data to the model;

ordering output data produced in response to a measurement of quality; and

adjusting operation of said model in response to a result of said ordering step.

23. A computer-executed process for training a neural network stored in a computer-related medium for predicting results on the basis of selected characteristics which comprises a plurality of intercoupled processing elements, each processing element comprising:

a plurality of inputs;

a plurality of weights related to corresponding inputs for generating weighted inputs;

means for combining said weighted inputs; and

a transfer function for processing said combined weighted inputs to generate an output;

wherein said process comprises the iterative steps of:

applying input data to the neural network;

ordering output data produced in response to a measurement of quality; and

adjusting operation of the neural network in response to a result of the ordering step.

24. The computer-executed process as claimed in claim 23, wherein the step of adjusting operation of the neural network comprises the substeps of:

selecting a subset of the weights to be decayed; and

decaying said selected subset of weights.

25. The computer-executed process as claimed in claim 24, wherein said substep of selecting a partial subset of the weights to be decayed comprises a step for applying and minimizing a cost function containing an interlayer gain multiplier that varies decay rate in response to the location of a weight within said network.

26. The computer-executed process as claimed in claim 25, wherein the cost function is represented by the

expression:

$$\frac{1}{2}\sum_{k \in D}(target_k - output_k)^2 + g\lambda\sum_{i \in w}(c_i w_i^2 - \frac{1}{1+|w_i|})$$

Here:

D expresses a data set;

$target_k$ expresses a target value for an element k of the data set;

$output_k$ expresses a network output for the element k of the data set;

g expresses the interlayer gain multiplier;

l expresses the relative importance of a complexity term;

w expresses a weight set;

w_i expresses a value of a weight i; and

c_i expresses a constant.

27. A system for detecting transaction fraud on a customer account, comprising:

a predictive model for determining a likelihood of transaction fraud;

past transaction data input means for obtaining past transaction data;

a model development constituent element coupled to said predictive model for training said predictive model for determining said likelihood of transaction fraud;

past transaction data input means for obtaining past transaction data;

a past transaction data pre-processor for deducing past fraud-related variables from said past transaction data;

a model development constituent element coupled to the predictive model for training said predictive model from said past fraud-related variables;

a storage device for storing said predictive model;

current transaction data input means for obtaining current transaction data related to said transaction;

a current transaction data pre-processor for deducing current fraud-related variables from said current transaction data and sending said current fraud-related variables to said predictive model;

customer data input means for obtaining customer data;

a customer data pre-processor for deriving customer fraud-related variables from said customer data and sending said customer fraud-related variables to said predictive model from said past transaction data;

a storage device for storing said trained predictive model;

current transaction data input means for obtaining current transaction data and sending said current transaction data to said predictive model;

customer data input means for obtaining customer data and sending said customer data to said predictive model; and

an output device coupled to said predictive model for generating a signal in response to likelihood of said fraud.

28. The system as claimed in claim 27, wherein said model development constituent element comprises a past transaction data pre-processor for deriving past fraud-related variables from past transaction data.

29. The system as claimed in claim 27, wherein the predictive model comprises a neural network.

30. A system for detecting transaction fraud in an account belonging to a customer, comprising:

a predictive model; and

an output device coupled to said predictive model for generating a signal in response to likelihood of said fraud.

31. The system as claimed in claim 29, wherein said predictive model comprises a neural network.

[Detailed Description of the Invention]

CROSS-REFERENCE OF THIS APPLICATION RELATED TO FRAUD DETECTION USING PREDICTIVE MODELING

The subject matter of this application is related to the subject matter of pending U.S. Patent Application No. 07/814,179 "Neural Network Having Expert System Functionality" by Curt A. Levey filed December 30, 1991, the disclosed contents of which are incorporated herein by reference.

BACKGROUND OF THE INVENTION

1. Technical Field of the Invention

This invention relates in general to the detection of fraudulent use of customer accounts and account

numbers including, for example, credit card transactions, and more particularly relates to an automated fraud detection system and method that, using a predictive model, performs a pattern recognition and classification in order to isolate transactions having a high likelihood of fraud.

2. Description of the Related Art

The term "credit card" is used in the following discussion for illustrative purposes. However, the techniques and principles discussed in this specification are applicable to other types of customer accounts such as charge cards, bank automated teller machine cards and telephone cards.

Credit card issuers have conventionally attempted to limit loss due to fraud by closing a customer account immediately upon receipt of a report that the card has been lost or stolen. The customer's credit information is subsequently transferred to a new account and a new card is issued. This procedure is effective in limiting the fraudulent use of lost or stolen cards only after the loss or theft has been reported to the issuer.

However, in many instances the cardholder is ignorant of the fraudulent use and, accordingly, often there is no report made to the issuer. There is a possibility of fraud being perpetrated if the customer is unaware that the card has been lost or stolen, or as a result of other techniques such as the manufacture of a counterfeit card, merchant fraud, application fraud, or credit card mail interception are employed. In all these cases, the fraudulent use may remain undetected until such time as the cardholder notices an unfamiliar transaction on their next monthly statement and decides to contest the corresponding charge. This delay in detection of fraud may result in significant monetary loss. User fraud, in which the user claims that a valid transaction as invalid, is also possible.

Issuers of credit cards have sought to limit loss by fraud by attempting to detect fraudulent use before the cardholder has reported a lost or stolen card. A representative example method thereof is parameter analysis. The parameter analysis fraud detection method is based on a decision made using a small number of database fields coupled using a simple Boolean condition. An example condition thereof is indicated below:

When (number of transactions in 24 hours > X) and (an amount greater than Y dollars has been authenticated), this account is flagged as high risk

Parameter analysis provides X and Y values that satisfy either a required detection rate or a required false positive rate. In a hypothetical example, parameter values of X=400 and Y=1000 are able to uncover 20% of frauds at a false positive rate of 200:1, while parameter values of X=6 and Y=2000 are able to uncover 8% of frauds at a false positive rate of 20:1.

While the rules provided by parameter analysis are restricted to combinations of Boolean calculations (e.g., and, or) that as a condition use single variables, they are easily implemented in a database management system.

The rules of parameter analysis are derived by testing of single variables by which fraudulent activity is most reliably able to be distinguished from non-fraudulent activity. Only single-variable threshold comparisons are used and, accordingly, complex interactions between variables cannot be ascertained. This limitation reduces the potential of the system to discriminate between fraudulent activity and proper account conduct and results in low detection rates and high false-positive rates.

Moreover, an effective fraud detection model

generally requires more variables than conventional parameter analysis systems can handle. Furthermore, parameter analysis systems need to be frequently redeveloped to uncover new fraud schemes, and the automated redevelopment of these systems is difficult.

To that end, an automated system for screening transactions and isolating those transactions for which there is a concern that they may be fraudulent and which, while maintaining a relatively low false-positive rate, is able to uncover a relatively high proportion of fraud is desirable. The system should preferably be able to handle a large number of interdependent variables, and should possess a function by which the fundamental system model thereof can be redeveloped in new patterns for ascertaining fraudulent activity.

SUMMARY OF THE INVENTION

The present invention, which uses a predictive model such as a neural network to estimate individual customer accounts and identify likelihood of transaction fraud based on learned relationships among known variables, provides an automated system and method for detecting fraudulent transactions. These relationships enable the system to estimate the probability of fraud for each transaction. This probability is either provided as output to a human decision-maker required for processing the transaction or, when the probability exceeds a predetermined amount, a signal is sent to the issuer. The system also outputs reason codes that indicate the contributions of various factors related to a particular result. Finally, the system periodically monitors its performance and redevelops the model when performance drops below a predetermined level.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a block diagram pertaining to the embodying of the present invention;

FIG. 2 is an example of a sample system monitor screen that forms part of a typical output interface of the present invention;

FIG. 3 is an example of an account selection screen that forms part of a typical output interface of the present invention;

FIG. 4 is an example of a transaction analysis screen that forms part of a typical output interface of the present invention;

FIG. 5 is an example of a customer information screen that forms part of a typical output interface of the present invention;

FIG. 6 is an example of an analyst response screen that forms part of a typical output interface of the present invention;

FIG. 7 is a flowchart showing the principal functions and operation of the present invention;

FIG. 8 is a block diagram showing the overall functional architecture of the present invention;

FIG. 9 is a diagram of a single processing element within a neural network;

FIG. 10 is a diagram of a hidden processing element in a neural network;

FIG. 11 is a flowchart of a pre-processing method of the present invention;

FIG. 12 is a flowchart of a method for producing a profile record of the present invention;

FIG. 13 is a flowchart of a method for updating a profile record of the present invention;

FIG. 14 is a flowchart showing the operation of a batch transaction processing system based on the present invention;

FIG. 15 is a flowchart showing the operation of a semi-real-time transaction processing system based on the present invention;

FIG. 16 is a flowchart showing the operation of a real-time processing system based on the present invention;

FIG. 17 is a flowchart showing the overall operation of a transaction processing constituent element of the present invention;

FIG. 18 is a flowchart showing the operation of a module CSCORE of the present invention;

FIG. 19 is a flowchart showing the operation of a DeployNet of the present invention;

FIG. 20 is a flowchart showing a cascade operation of the present invention; and

FIG. 21 is a portion of a typical CFG model definition file.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

These drawings are used for the purpose of illustration alone and depict preferred embodiments of the present invention. As will be readily understood by those skilled in the art from the following discussion, alternative embodiments of the structures and methods described in this specification may be employed without departing from the principles of the invention described herein.

FIG. 1 is a block diagram of a system 100 serving as a typical embodiment based on the present invention. Transaction information is applied to the system 100 via data network 105 connected to a conventional

financial data facility 106 where transaction information is collected from conventional sources such as a human-operated credit-card authentication terminal and an automated teller machine (not shown). A CPU 101 executes a software program instruction stored in a program storage region 107, and this instruction directs the CPU 101 to execute the various functions of the system. In a preferred embodiment thereof, the software program is written in the ANSI C language that may be executed on a variety of conventional hardware platforms. In accordance with the software program instruction, the CPU 101 stores data obtained from a data network 105 in a data storage region 103, and uses a RAM 102 as an operating region in a conventional manner. The CPU 101, the data storage region 103 and the program storage region 107 operate together to provide a neural network model 108 for predicting fraud. The neural network 108 processes this information and, as described below, in order to obtain an indication of the likelihood of fraud, a signal indicating the likelihood thereof is sent from the CPU 101 to an output device 104.

In a preferred embodiment, the CPU 101 is a IBM mainframe computer 3090 model, the RAM 102 and the data storage region 103 are conventional RAM, ROM and disk storage devices for the Model 3090 CPU, and the output device 104 constitutes conventional averages for either printing a result based on a signal generated by the neural network 108 or displaying a result on a video screen using a window-based interface system, or sending a result to a database for later access or sending a signal dependent on this result to an authentication system (not shown in the diagram) for further processing.

FIG. 2 to FIG. 6 show examples of screens from a conventional window-based interface system (not shown in the diagram) that forms part of the output device 104. FIG. 2 shows a system monitor 201 that allows a

fraud analyst or system supervisor to monitor system performance. The system monitor 201 shows a cutoff score 202 (an account above this score is flagged), a number of accounts with scores above the cutoff 203, a fraud score 204 related to a specific account, and an account number 205.

FIG. 3 shows an account selection screen 301 that includes a scrolling window 302 by which an analyst, by monitoring, can select high-risk transactions, and a series of buttons 303 that allow the analyst to select further operations related to the selected transaction.

FIG. 4 shows a transaction analysis screen 401 by which a fraud analyst can examine each high-risk transaction and determine an appropriate fraud control action. This screen includes account information 402, fraud score 403, explanations deduced from reason codes 404 that indicate reasons related to the fraud score 403, and two scrolling windows 405 and 406 that show transaction information for the current day and the last seven days 405, and for the last six months 406.

FIG. 5 shows a customer information screen 501 by which an analyst can access customer information, the screen including account number 502, customer name 503, best time to call 504, phone number 505 and address 506. The screen also provides access to additional functions via on-screen buttons 507.

FIG. 6 shows an analyst response screen 601 by which an analyst can record actions implemented to control fraud. The screen includes a series of check boxes 602 for recording information, a comment box 603, and on-screen buttons 604 allowing access to other functions.

FIG. 7 is an overall flowchart of the principal functions and operation of the system 100. A first neural network model 108 is trained (701) using data that describes past transactions from the data network

105. The data describing the network model is then stored (702). Once the model description is stored, the system 100 is able to process current transactions. The system 100 obtains data related to current transactions (703) and applies the current transaction data to the stored network model (704). The model 108 determines a fraud score and a reason code (described below) that are output (705) to the user or to a database, or to another system via the output device 104.

FIG. 8 shows the overall functional architecture of the system 100. The system 100 is sorted into two major constituent elements, that is to say, a model development constituent element 801 and a transaction processing constituent element 802. The model development constituent element 801 employs past data to construct the neural network 108 containing the information expressing learned relationships among a number of variables. Together, the learned relationships form a behavior model of the variables. While a neural network is used in the preferred embodiment, any type of predictive modeling technique may be used. For purposes of illustration, the present invention is described in this specification using the term neural network.

A transaction processing constituent element 802 executes the following three functions. 1) The transaction processing constituent element 802 determines the likelihood of fraud for each transaction by supplying data from various information sources 805, 806 into the neural network 108, obtaining results, and outputting these results to 807. 2) When applicable, the transaction processing constituent element 802 produces a record in a profile database 806 summarizing past customer transaction patterns. 3) When applicable, the transaction processing constituent element 802 updates the appropriate records of the profile database 806.

The two constituent elements of the system will be

described in turn.

Model Development constituent element 801

Neural Networks: Neural networks employ a technique of "learning" relationships based on repeated exposure to data and internal weight adjustment. Neural networks facilitate rapid model development and automated data analysis. Fundamentally, these networks express a static modeling technique by which models can be constructed from data containing both linear and non-linear relationships. While similar in concept to regression analysis, nonlinearity and interactions of independent variables can be ascertained using neural networks that do not need to be described in advance. In other words, while conventional regression analysis necessitates a manual detection and specifying of nonlinearities and interactions, neural networks execute these operations automatically. For a more detailed description of neural networks see D.E. Rumelhart et al., "Learning Representations by Back-Propagating Errors", Nature v. 323, pp. 533-36 (1986), and R. Hecht-Nielsen, "Theory of the Backpropagation Neural Network", Neural Networks for Perception, pp. 65-93 (1992). The teachings thereof are incorporated herein by reference.

Neural networks are configured from a large number of intercoupled neuron-like processing elements that send data to each other along these connections. The strength of the connections between these processing elements is expressed by weights. FIG. 9 shows a diagram of a single processing element 901. The processing element receives inputs X_1, X_2, ... X_n from other processing elements or directly from inputs to the system. The processing element multiplies each of these inputs with an associated weight W_1, W_2, ... W_n and adds the results thereof to form a weighted sum 902. The processing element then applies a transfer function 903 (typically non-linear) to the weighted sum to obtain a value Z known as the element state. This

state Z is then either passed on to another element along a weighted connection, or is provided as an output signal. The states are collectively employed to represent information across a short term, while weights represent long-term information or learning.

The processing elements in a neural network can be grouped into three categories. That is to say, input processing elements (elements which receive input data values), output processing elements (elements which produce output values), and hidden processing elements (all other elements). The purpose of a hidden processing element is to allow the neural network to build intermediate representations in which input data is combined in a state that facilitates a model learning the desired mapping with greater accuracy. FIG. 10 is a diagram showing the concept of a hidden processing element. Inputs 1001 are supplied to a layer of input processing elements 1002. The outputs of the input elements are passed to a hidden element layer 1003. Normally, there are several layers of hidden elements. Eventually, the hidden elements transfer outputs to an output element layer 1004, and the output elements generate an output value 1005.

Neural networks learn from examples by modifying these weights. The "training" process, of which the general techniques are known to those skilled in the art, includes the following steps:

1) Repeatedly supplying examples of a particular input/output operation to the neural network model;

2) Comparing a model output with a desired output to measure error; and

3) Modifying model weights to reduce error.

This series of steps is repeated until such time as this iteration no longer decreases the error. Subsequently, the network is said to have been "trained". Once training is completed, the network can

predict outcomes for new data inputs.

Fraud-Related Variables

Data employed to train the model of the present invention is supplied from various database files containing time-based data on individual transactions, merchants, and customers. These data are desirably pre-processed before being supplied to the neural network and, as a result, desirably produce several subsets of fraud-related variables that have been empirically determined to be more effective predictors of fraud than the original historical data.

FIG. 11 shows a flowchart of the pre-processing method of the present invention. Individual elements of the flowchart are designated by names that relate to module names.

The data employed for pre-processing is supplied from the following three databases containing past data: 1) Past transaction database 1101 (also known as an "authentication database") containing past transaction data of the past two years that is able to be executed in the same database as the past data 804; 2) Customer database 1103 containing customer data; and 3) Fraud database 1102 that indicates in which account fraudulent activity has occurred and when the fraudulent activity occurred.

A module readauth.sas 1104 reads transaction data from the past transaction database 1101. A module matchauth.sas 1105 samples this transaction data to obtain a new transaction data set containing all of the fraud accounts and a randomly selected subset of the non-fraud accounts. While the new transaction data set is being produced, module matchauth.sas 1105 uses information from the fraud database 1102 to determine in which accounts fraud has occurred and in which accounts no fraud has occurred. From the viewpoint of effective network training, it has been found

preferable to obtain approximately ten non-fraud accounts per fraud account.

A module readex.sas 1106 reads customer data from the customer database 1103. A module matchex.sas 1107 samples this customer data to obtain a new customer data set containing all of the fraud accounts and the same non-fraud account subsets obtained by the module matchauth.sas. While the new customer data set is being produced, the module matchex.sas 1107 uses information from the fraud database 1102 to determine in which accounts fraud has occurred and in which accounts no fraud has occurred.

A module mxmerge.sas 1108 merges all of the data sets obtained by modules matchauth.sas 1105 and matchex.sas 1107. A module genau.sas 1109 subdivides the merged data set into subsets of monthly data.

A module gensamp.sas 1112 samples the data set created by the module mxmerge.sas 1108 and subdivided by genau.sas 1109, and creates a new data set known as sample.ssd where each record is represented with transaction activity of a specific account on a specific day. A module gensamp.sas 1112 uses information from the fraud database 1102 to determine which records are fraudulent. The module gensamp.sas 1112 provides the following subset of authentication days: a set of valid account-days generated by removing a plurality of transactions for the same customer on a same day from a database of all transactions. Each account day of the set of active account-days is assigned a draft number from 0 to 1. The draft numbers are assigned as follows: if the account-day is non-fraudulent the draft number is set to a random number between 0 and 1. If the account-day is fraudulent and falls on a first or second day of fraud the draft number is set to 0. In other instances it is set to 1. 25,000 account-days of smallest draft number are then selected for inclusion in a sample.ssd. Accordingly, all fraudulent account-day (up to 25,000) and non-

fraudulent account-day queues are included in the sample.ssd.

A module roll15.sas 1113 generates a 15-day rolling window of data. This data comprises a plurality of multiple records for each account-day listed in the sample.ssd. The current day and the 14 preceding days are listed in each sample account.

A module roll15to7.sas 1117 uses a roll15 data set and filters out days 8 to 15 to produce a roll7, that is to say, a 7-day rolling window data set 1119. Days 8 to 15 are ignored. A module genrolv.sas 1118 generates input variables for a rolling window of the previous 15 days of transactions. The module processes a data set comprising a plurality of records per account of which the number is variable, and produces a data set comprising one record per account. The result is referred to as the rollv.ssd.

A module roll15to1.sas 1114 uses the roll15 data set and filters all except the current day to produce roll1. A module gencurv.sas 1115 uses the roll1 to generate current day variables 1116 that last transactions occurring during the current day.

A module genprof.sas generates (profile) variables that form (profile) records 1111.

A module merge.sas 1120 combines the (profile) records 1111, the 1-day variables 1116 and the 7- day variables 1119 and generates, from the combined result thereof, new fraud-related variables as listed below. The module also merges the rollv.ssd with the (profile) data sets containing the filtered samples to produce a single data set comprising both (profile) and rolling window variables. The result, which is referred to as a modln2 data set 1121 (also referred to as a "training set"), contains the fraud-related variables needed to train the network. A module Scaler 1122 scales the variables of the scaled training set so that the

averages value thereof is 0.0 and the standard deviation is 1.0 generating a scaled modln2 data set 1123.

Most fraud-related variables are able to be generated employing variables based on the pre-processing method described above. The following fraud-related variables are used in a preferred embodiment thereof:

• Customer use pattern (profiles) expressing the time-of-day and the day-of-week

• Credit card expiry date

• Dollar amount spent in each SIC (Standard Industrial Classification) merchant group category in a current day;

• Percentage of dollars spent by a customer in each SIC merchant group category in a current day

• Number of transactions in each SIC merchant group category in a current day

• Percentage of number of transactions in each SIC merchant group category in a (current day)

• Categorization of SIC merchant group category by fraud rate (high, medium, or low risk)

• Categorization of SIC merchant group category by customer type (customer groups that most frequently use a certain SIC category)

• Categorization of geographical regions by fraud rate (high, medium, or low risk)

• Categorization of geographical region by customer type

• Average number of days between transactions

• Variance of number of days between transactions

• Average time between transactions in one day

• Variance of time between transactions in one day

• Number of multiple transaction declines at same merchant

• Number of out-of-state transactions

• Average number of transaction declines

• Year-to-date high balance

• Transaction amount

• Transaction date and time

• Transaction type

Other supplementary fraud-related variables that may be considered are listed below:

(Current day) Cardholder Fraud-Related Variables

bweekend - (current day) Boolean variable indicating a current date and time deemed as being the weekend

cavapvdl - (current day)average approved dollar amount

cavaudl - (current day) average authenticated dollars per day

ccoscdom - (current day) cosine function of day of month, i.e. cos(day((datepart(cst_dt) *&TWOPI)/30))

ccoscdow - (current day) cosine function of day of week, i.e. cos(weekday((datepart(cst_dt) *&TWOPI)/7))

ccoscmoy - (current day) cosine function of month of year, i.e. cos(month((datepart(cst_dt) *&TWOPI)/12))

cdom - (current day) day of month

cdow - (current day) day of week

chdzip code - (current day) cardholder zip code

chibal - (current day) high balance

chidcapv - (current day) highest dollar amount of single cash approval

chidcdec - (current day) highest dollar amount of single cash decline

chidmapv - (current day) highest dollar amount of single merchandise approval

chidmdec - (current day) highest dollar amount of single purchase decline

chidsapv - (current day) highest dollar amount of single approval

chidsau - (current day) highest dollar amount of single authentication

chidsdec - (current day) highest dollar amount of single decline

cmoy - (current day) month of year

cratdcau - (current day) ratio of declines to authentications

csincdom - (current day) sine function of day of month, i.e. sin(day ((datepart(cst_dt) *&TWOPI)/30))

csincdow - (current day) sine function of day of week, i.e. sin(weekday((datepart(cst_dt) *&TWOPI)/7))

csincmoy - (current day) sine function of month of year, i.e. sin(month ((datepart(cst_dt) *&TWOPI)/12))

cst_dt - (current day) cst datetime deduced from zip code and CST authenticated time

ctdapv - (current day) total dollar approvals

ctdau - (current day) total dollar authentications

ctdcsapv - (current day) total dollar cash advance approvals

ctdcsdec - (current day) total dollar cash advance

declines

ctddec - (current day) total dollar declines

ctdmrapv - (current day) total dollar merchandise approvals

ctdmrdec - (current day) total dollar merchandise declines

ctnapv - (current day) total number of approvals

ctnau - (current day) total number of authentications

ctnaul0d - (current day) number of authentications in day<=$10

ctnaudy - (current day) total number of authentications in a day

ctncsapv - (current day) total number of cash advance approvals

ctncsapv - (current day) total number of cash approvals

ctncsdec - (current day) total number of cash advance declines

ctndec - (current day) total number of declines

ctnmrapv - (current day) total number of merchandise approvals

ctnmrdec - (current day) total number of merchandise declines

ctnsdapv - (current day) total number of approvals on same day of week as current day

ctnwdaft - (current day) total number of weekday afternoon approvals

ctnwdapv - (current day) total number of weekday approvals

ctnwdeve - (current day) total number of weekday evening approvals

ctnwdmor - (current day) total number of weekday morning approvals

ctnwdnit - (current day) total number of weekday night approvals

ctnweaft - (current day) total number of weekend afternoon approvals

ctnweapv (current day) total number of weekend approvals

ctnweeve (current day) total number of weekend evening approvals

ctnwemor (current day) total number of weekend morning approvals

ctnwenit (current day) total number of weekend night approvals

currbal (current day) current balance

cvrandl (current day) variance of dollars per authentication in a day

czrate1 (current day) zip code risk group 1 'Zip code very high fraud rate'

czrate2 (current day) zip code risk group 2 'Zip code high fraud rate'

czrate3 (current day) zip code risk group 3 'Zip code medium high fraud rate'

czrate4 (current day) zip code risk group 4 'Zip code medium fraud rate'

czrate5 (current day) zip code risk group 5 'Zip code medium low fraud rate'

czrate6 (current day) zip code risk group 6 'Zip code low fraud rate'

czrate7 (current day) zip code risk group 7 'Zip code very low fraud rate'

czrate8 (current day) zip code risk group 8 'Zip code unknown fraud rate'

ctdsfa01 (current day) total dollars of transactions in SIC factor group 01

ctdsfa02 (current day) total dollars of transactions in SIC factor group 02

ctdsfa03 (current day) total dollars of transactions in SIC factor group 03

ctdsfa04 (current day) total dollars of transactions in SIC factor group 04

ctdsfa05 (current day) total dollars of transactions in SIC factor group 05

ctdsfa06 (current day) total dollars of transactions in SIC factor group 06

ctdsfa07 (current day) total dollars of transactions in SIC factor group 07

ctdsfa08 (current day) total dollars of transactions in SIC factor group 08

ctdsfa09 (current day) total dollars of transactions in SIC factor group 09

ctdsfa10 (current day) total dollars of transactions in SIC factor group 10

ctdsfa11 (current day) total dollars of transactions in SIC factor group 11

ctdsra01 (current day) total dollars of transactions in SIC fraud rate group 01

ctdsra02 (current day) total dollars of transactions in SIC fraud rate group 02

ctdsra03 (current day) total dollars of transactions in SIC fraud rate group 03

ctdsra04 (current day) total dollars of transactions in

SIC fraud rate group 04

ctdsra05 (current day) total dollars of transactions in SIC fraud rate group 05

ctdsra06 (current day) total dollars of transactions in SIC fraud rate group 06

ctdsra07 (current day) total dollars of transactions in SIC fraud rate group 07

ctdsva01 (current day) total dollars in SIC VISA group 01

ctdsva02 (current day) total dollars in SIC VISA group 02

ctdsva03 (current day) total dollars in SIC VISA group 03

ctdsva04 (current day) total dollars in SIC VISA group 04

ctdsva05 (current day) total dollars in SIC VISA group 05

ctdsva06 (current day) total dollars in SIC VISA group 06

ctdsva07 (current day) total dollars in SIC VISA group 07

ctdsva08 (current day) total dollars in SIC VISA group 08

ctdsva09 (current day) total dollars in SIC VISA group 09

ctdsva10 (current day) total dollars in SIC VISA group 10

ctdsva11 (current day) total dollars in SIC VISA group 11

ctnsfa01 (current day) total number of transactions in SIC factor group 01

ctnsfa02 (current day) total number of transactions in SIC factor group 02

ctnsfa03 (current day) total number of transactions in SIC factor group 03

ctnsfa04 (current day) total number of transactions in SIC factor group 04

ctnsfa05 (current day) total number of transactions in SIC factor group 05

ctnsfa06 (current day) total number of transactions in SIC factor group 06

ctnsfa07 (current day) total number of transactions in SIC factor group 07

ctnsfa08 (current day) total number of transactions in SIC factor group 08

ctnsfa09 (current day) total number of transactions in SIC factor group 09

ctnsfa10 (current day) total number of transactions in SIC factor group 10

ctnsfa11 (current day) total number of transactions in SIC factor group 11

ctnsra01 (current day) total number of transactions in SIC fraud rate group 01

ctnsra02 (current day) total number of transactions in SIC fraud rate group 02

ctnsra03 (current day) total number of transactions in SIC fraud rate group 03

ctnsra04 (current day) total number of transactions in SIC fraud rate group 04

ctnsra05 (current day) total number of transactions in SIC fraud rate group 05

ctnsra06 (current day) total number of transactions in

SIC fraud rate group 06

ctnsra07 (current day) total number of transactions in SIC fraud rate group 07

ctnsva01 (current day) total number of SIC VISA group 01

ctnsva02 (current day) total number of SIC VISA group 02

ctnsva03 (current day) total number of SIC VISA group 03

ctnsva04 (current day) total number of SIC VISA group 04

ctnsva05 (current day) total number of SIC VISA group 05

ctnsva06 (current day) total number of SIC VISA group 06

ctnsva07 (current day) total number of SIC VISA group 07

ctnsva08 (current day) total number of SIC VISA group 08

ctnsva09 (current day) total number of SIC VISA group 09

ctnsva10 (current day) total number of SIC VISA group 10

ctnsva11 (current day) total number of SIC VISA group 11

(7-Day) Cardholder Fraud-Related Variables

raudymdy (7-day) ratio of authentication days to number of days in window

ravapvdl (7-day) average dollar amount for authentication

ravaudl (7-day) average dollars per authentication in a window

rddapv (7-day) average dollars per day of approvals

rddapv2 (7-day) average dollars per day of approvals on days on which there is authentication

rddau (7-day) average dollars per day of authentications on days on which there is authentication

rddauall (7-day) average dollars per day of authentications on all days in window

rddcsapv (7-day) average dollars per day of cash approvals

rddcsdec (7-day) average dollars per day of cash declines

rdddec (7-day) average dollars per day of declines

rdddec2 (7-day) average dollars per day of declines on days on which there is authentication

rddmrapv (7-day) average dollars per day of merchandise approvals

rddmrdec (7-day) average dollars per day of merchandise declines

rdnapv (7-day) average number per day of approvals

rdnau (7-day) average number per day of authentications on days on which there is authentication

rdnauall (7-day) average number per day of authentications on all days of a window

rdncsapv (7-day) average number per day of cash approvals

rdncsdec (7-day) average number per day of cash declines

rdndec (7-day) average number per day of declines

rdnmrapv (7-day) average number per day of merchandise approvals

rdnmrdec (7-day) average number per day of merchandise declines

rdnsdap2 (7-day) average number per day of approvals on the same day of the week as the approvals as calculated for approved days only

rdnsdapv (7-day) average number per day of approvals on the same day of the week as current day

rdnwdaft (7-day) average number per day of weekday afternoon approvals

rdnwdapv (7-day) average number per day of weekday approvals

rdnwdeve (7-day) average number per day of weekday evening approvals

rdnwdmor (7-day) average number per day of weekday morning approvals

rdnwdnit (7-day) average number per day of weekday night approvals

rdnweaft (7-day) average number per day of weekend afternoon approvals

rdnweapv (7-day) average number per day of weekend approvals

rdnweeve (7-day) average number per day of weekend evening approvals

rdnwemor (7-day) average number per day of weekend morning approvals

rdnwenit (7-day) average number per day of weekend night approvals

rhibal (7-day) highest balance of window

rhidcapv (7-day) highest dollar amount of single cash approval

rhidcdec (7-day) highest dollar amount of single cash decline

rhidmapv (7-day) highest dollar amount of single merchandise approval

rhidmdec (7-day) highest dollar amount of single merchandise decline

rhidsapv (7-day) highest dollar amount of single approval

rhidsau (7-day) highest dollar amount of single authentication

rhidsdec (7-day) highest dollar amount of single decline

rhidtapv (7-day) highest total dollar amount of an approval in a single day

rhidtau (7-day) highest total dollar amount of an authentication in a single day

rhidtdec (7-day) highest total dollar amount of a decline in a single day

rhinapv (7-day) highest number of approvals in a single day

rhinau (7-day) highest number of authentications in a single day

rhindec (7-day) highest number of declines in a single day

rnaudy (7-day) number of days in any authenticated window

rnausd (7-day) number of days in any authenticated same day of week

rnauwd (7-day) number of weekdays in any authenticated

window

rnauwe (7-day) number of weekend days in any authenticated window

rncsandy (7-day) number of days in cash authenticated window

rnmraudy (7-day) number of days in merchandise authenticated window

rtdapv (7-day) total dollars of approvals

rtdau (7-day) total dollars of authentications

rtdcsapv (7-day) total dollars of cash advance approvals

rtdcsdec (7-day) total dollars of cash advance declines

rtddec (7-day) total dollars of declines

rtdmrapv (7-day) total dollars of merchandise approvals

rtdmrdec (7-day) total dollars of merchandise declines

rtnapv (7-day) total number of approvals

rtnapvdy (7-day) total number of approvals in a day

rtnau (7-day) total number of authentications

rtnaul0d (7-day) number of authentications in a window of authentication <=$10

rtncsapv (7-day) total number of cash advance approvals

rtncsdec (7-day) total number of cash advance declines

rtndec (7-day) total number of declines

rtnmrapv (7-day) total number of merchandise approvals

rtnmrdec (7-day) total number of merchandise declines

rtnsdapv (7-day) total number of approvals on same day of week as current day

rtnwdaft (7-day) total number of weekday afternoon

approvals

rtnwdapv (7-day) total number of weekday approvals

rtnwdeve (7-day) total number of weekday evening approvals

rtnwdmor (7-day) total number of weekday morning approvals

rtnwdnit (7-day) total number of weekday night approvals

rtnweaft (7-day) total number of weekend afternoon approvals

rtnweapv (7-day) total number of weekend approvals

rtnweeve (7-day) total number of weekend evening approvals

rtnwemor (7-day) total number of weekend morning approvals

rtnwenit (7-day) total number of weekend night approvals

rvraudl (7-day) variance of dollars per authentication in a window

(Profile) Cardholder Fraud-Related Variables

paudymdy - (profile) ratio of authentication days to number of days in the month

pavapvdl - (profile) average dollar amount for authentication

pavaudl - (profile) average dollars per authentication each month

pchdzip code - (profile) final number of zip code of the cardholder

pdbm - (profile) value of 'date became member' at time of final profile update

pddapv - (profile) average of dollar approvals

pddapv2 - (profile) daily average dollars of approvals on days on which there is authentication

pddau - (profile) daily average dollars of authentications on days on which there is authentication

pddau30 - (profile) daily average dollars of authentications on all days in month

pddcsapv - (profile) daily average dollars of cash approvals

pddcsdec - (profile) daily average dollars of cash declines

pddde - (profile) daily average dollars of declines

pddde2 - (profile) daily average dollars of declines on days on which there is authentication

pddmrapv - (profile) daily average dollars of merchandise approvals

pddmrdec - (profile) daily average dollars of merchandise declines

pdnapv - (profile) daily average number of approvals

pdnau - (profile) daily average number of authentications on days on which there is authentication

pdnau30 - (profile) daily average number of authentications on all days in month

pdncsapv - (profile) daily average number of cash approvals

pdncsdec - (profile) daily average number of cash declines

pdndec - (profile) daily average number of declines

pdnmrapv - (profile) daily average number of merchandise approvals

pdnmrdec - (profile) daily average number of merchandise declines

pdnw1ap2 - (profile) average number of approvals on authenticated Sunday

pdnw1apv - (profile) average number of approvals on Sunday (1st day of week)

pdnw2ap2 - (profile) average number of approvals on Mondays on which there is authentication

pdnw2apv - (profile) average number of approvals on Monday (2nd day of week)

pdnw3ap2 - (profile) average number of approvals on Tuesdays on which there is authentication

pdnw3apv - (profile) average number of approvals on Tuesday (3rd day of week)

pdnw4ap2 - (profile) average number of approvals on Wednesdays on which there is authentication

pdnw4apv - (profile) average number of approvals on Wednesday (4th day of week)

pdnw5ap2 - (profile) average number of approvals on Thursdays on which there is authentication

pdnw5apv - (profile) average number of approvals on Thursday (5th day of week)

pdnw6ap2 - (profile) average number of approvals on Fridays on which there is authentication

pdnw6apv - (profile) average number of approvals on Friday (6th day of week)

pdnw7ap2 - (profile) average number of approvals on Saturdays on which there is authentication

pdnw7apv - (profile) average number of approvals on

Saturdays (7th day of week)

pdnwdaft - (profile) daily average number of weekday afternoon approvals

pdnwdapv - (profile) daily average number of weekday approvals

pdnwdeve - (profile) daily average number of weekday evening approvals

pdnwdmor - (profile) daily average number of weekday morning approvals

pdnwdnit - (profile) daily average number of weekday night approvals

pdnweaft - (profile) daily average number of weekend afternoon approvals

pdnweapv - (profile) daily average number of weekend approvals

pdnweeve - (profile) daily average number of weekend evening approvals

pdnwemor - (profile) daily average number of weekend morning approvals

pdnwenit - (profile) daily average number of weekend night approvals

pexpir - (profile) expiry date stored in profile; updated if current day > expiry date

phibal - (profile) highest monthly balance

phidcapv - (profile) highest dollar amount of a single cash approval in a month

phidcdec - (profile) highest dollar amount of a single cash decline in a month

phidmapv - (profile) highest dollar amount of a single merchandise approval in a month

phidmdec - (profile) highest dollar amount of a single merchandise decline in a month

phidsapv - (profile) highest dollar amount of a single approval in a month

phidsau - (profile) highest dollar amount of a single authentication in a month

phidsdec - (profile) highest dollar amount of a single decline in a month

phidtapv - (profile) highest total dollar amount for an approval in a single day

phidtau - (profile) highest total dollar amount of an authentication in a single day

phidtdec - (profile) highest total dollar amount of a decline in a single day

phinapv - (profile) highest number of approvals in a single day

phinau - (profile) highest number of authentications in a single day

phindec - (profile) highest number of declines in a single day

pm1avbal - (profile) average balance during first 10 days of month

pm1nauths - (profile) number of authentications in the first 10 days of month

pm2avbal - (profile) average bal. during 2nd 10 days of mo.

pm2nauths - (profile) number of authentications in the second 10 days of month

pm3avbal - (profile) average bal. during remaining days

pm3nauths - (profile) number of authentications in the last part of the month

pmovewt - (profile) final number of zip code used to determine recent change of residence; pmovewt=2 for move within the previous calendar month; pmovew

pnaudy - (profile) number of days on which there is authentication

pnauw1 - (profile) number of Sundays in month having any authentication

pnauw2 - (profile) number of Mondays in month having any authentication

pnauw3 - (profile) number of Tuesdays in month having any authentication

pnauw4 - (profile) number of Wednesdays in month having any authentication

pnauw5 - (profile) number of Thursdays in month having any authentication

pnauw6 - (profile) number of Fridays in month having any authentication

pnauw7 - (profile) number of Saturdays in month having any authentication

pnauwd - (profile) number of weekdays in month having any authentication

pnauwe - (profile) number of weekend days in month having any authentication

pncsaudy - (profile) number of days in month having cash authentication

pnmraudy - (profile) number of days in month having merchandise authentication

pnweekday - (profile) number of weekdays in month

pnweekend - (profile) number of weekend days in month

pratdcau - (profile) ratio of declines to authentications

profage - (profile) number of months account has had a (profile) (up to 6 months)

psdaudy - (profile) standard deviation of number of days between transactions in a month

psddau - (profile) standard deviation of dollars per authentication in a month

ptdapv - (profile) total dollars of approvals in a month

ptdau - (profile) total dollars of authentications in a month

ptdaudy - (profile) total dollars of authentications in a day

ptdcsapv - (profile) total dollars of cash advance approvals in a month

ptdcsdec - (profile) total dollars of cash advance declines in a month

ptddec - (profile) total dollars of declines in a month

ptdmrapv - (profile) total dollars of merchandise approvals in a month

ptdmrdec - (profile) total dollars of merchandise declines in a month

ptdsfa01 - (profile) total dollars of transactions in SIC factor group 01

ptdsfa02 - (profile) total dollars of transactions in SIC factor group 02

ptdsfa03 - (profile) total dollars of transactions in SIC factor group 03

ptdsfa04 - (profile) total dollars of transactions in SIC factor group 04

ptdsfa05 - (profile) total dollars of transactions in SIC factor group 05

ptdsfa06 - (profile) total dollars of transactions in SIC factor group 06

ptdsfa07 - (profile) total dollars of transactions in SIC factor group 07

ptdsfa08 - (profile) total dollars of transactions in SIC factor group 08

ptdsfa09 - (profile) total dollars of transactions in SIC factor group 09

ptdsfa10 - (profile) total dollars of transactions in SIC factor group 10

ptdsfa11 - (profile) total dollars of transactions in SIC factor group 11

ptdsra01 - (profile) total dollars of transactions in SIC fraud rate group 01

ptdsra02 - (profile) total dollars of transactions in SIC fraud rate group 02

ptdsra03 - (profile) total dollars of transactions in SIC fraud rate group 03

ptdsra04 - (profile) total dollars of transactions in SIC fraud rate group 04

ptdsra05 - (profile) total dollars of transactions in SIC fraud rate group 05

ptdsra06 - (profile) total dollars of transactions in SIC fraud rate group 06

ptdsra07 - (profile) total dollars of transactions in SIC fraud rate group 07

ptdsva01 - (profile) total dollars in SIC VISA group 01

ptdsva02 - (profile) total dollars in SIC VISA group 02

ptdsva03 - (profile) total dollars in SIC VISA group 03

ptdsva04 - (profile) total dollars in SIC VISA group 04

ptdsva05 - (profile) total dollars in SIC VISA group 05

ptdsva06 - (profile) total dollars in SIC VISA group 06

ptdsva07 - (profile) total dollars in SIC VISA group 07

ptdsva08 - (profile) total dollars in SIC VISA group 08

ptdsva09 - (profile) total dollars in SIC VISA group 09

ptdsva10 - (profile) total dollars in SIC VISA group 10

ptdsva11 - (profile) total dollars in SIC VISA group 11

ptnapv - (profile) total number of approvals in a month

ptnapvdy - (profile) total number of approvals a day

ptnau - (profile) total number of authentications in a month

ptnau10d - (profile) number of authentications in a month in which authentication<=$10

ptnaudy - (profile) total number of authentications in a day

ptncsapv - (profile) total number of cash advance approvals in a month

ptncsdec - (profile) total number of cash advance declines in a month

ptndec - (profile) total number of declines in a month

ptndecdy - (profile) total number of declines in a day

ptnmrapv - (profile) total number of merchandise approvals in a month

ptnmrdec - (profile) total number of merchandise declines in a month

ptnsfa01 - (profile) total number of transactions in SIC factor group 01

ptnsfa02 - (profile) total number of transactions in SIC factor group 02

ptnsfa03 - (profile) total number of transactions in SIC factor group 03

ptnsfa04 - (profile) total number of transactions in SIC factor group 04

ptnsfa05 - (profile) total number of transactions in SIC factor group 05

ptnsfa06 - (profile) total number of transactions in SIC factor group 06

ptnsfa07 - (profile) total number of transactions in SIC factor group 07

ptnsfa08 - (profile) total number of transactions in SIC factor group 08

ptnsfa09 - (profile) total number of transactions in SIC factor group 09

ptnsfa10 - (profile) total number of transactions in SIC factor group 10

ptnsfa11 - (profile) total number of transactions in SIC factor group 11

ptnsra01 - (profile) total number of transactions in SIC fraud rate group 01

ptnsra02 - (profile) total number of transactions in SIC fraud rate group 02

ptnsra03 - (profile) total number of transactions in SIC fraud rate group 03

ptnsra04 - (profile) total number of transactions in SIC fraud rate group 04

ptnsra05 - (profile) total number of transactions in SIC fraud rate group 05

ptnsra06 - (profile) total number of transactions in SIC fraud rate group 06

ptnsra07 - (profile) total number of transactions in

SIC fraud rate group 07

ptnsva01 - (profile) total number in SIC VISA group 01

ptnsva02 - (profile) total number in SIC VISA group 02

ptnsva03 - (profile) total number in SIC VISA group 03

ptnsva04 - (profile) total number in SIC VISA group 04

ptnsva05 - (profile) total number in SIC VISA group 05

ptnsva06 - (profile) total number in SIC VISA group 06

ptnsva07 - (profile) total number in SIC VISA group 07

ptnsva08 - (profile) total number in SIC VISA group 08

ptnsva09 - (profile) total number in SIC VISA group 09

ptnsva10 - (profile) total number in SIC VISA group 10

ptnsva11 - (profile) total number in SIC VISA group 11

ptnw1apv - (profile) total number of approvals on Sundays (1st day of week)

ptnw2apv - (profile) total number of approvals on Mondays (2nd day of week)

ptnw3apv - (profile) total number of approvals on Tuesdays (3rd day of week)

ptnw4apv - (profile) total number of approvals on Wednesdays (4th day of week)

ptnw5apv - (profile) total number of approvals on Thursdays (5th day of week)

ptnw6apv - (profile) total number of approvals on Fridays (6th day of week)

ptnw7apv - (profile) total number of approvals on Saturdays (7th day of week)

ptnwdaft - (profile) total number of weekday afternoon approvals in a month

ptnwdapv - (profile) total number of weekday approvals in a month

ptnwdeve - (profile) total number of weekday evening approvals in a month

ptnwdmor - (profile) total number of weekday morning approvals in a month

ptnwdnit - (profile) total number of weekday night approvals in a month

ptnweaft - (profile) total number of weekend afternoon approvals in a month

ptnweapv - (profile) total number of weekend approvals in a month

ptnweeve - (profile) total number of weekend evening approvals in a month

ptnwemor - (profile) total number of weekend morning approvals in a month

ptnwenit - (profile) total number of weekend night approvals in a month

pvdaybtwn - (profile) variance in number of days between transactions (number of days having minimum of 3 transactions)

pvraudl - (profile) variance of dollars per authentication in a month

(Merchant) Fraud Variables

mtotturn (Merchant) Total turnover for a specified merchant

msicturn (Merchant) SIC code cumulative turnover

mctrtage (Merchant) Age of contract for specified merchant

maagsic (Merchant) Average contract age for this SIC code

mavgnbtc (Merchant) Average number of batch transactions

maamttrx (Merchant) Average amount per transaction (average amount per authentication)

mvaramt (Merchant) Variance of amount per transaction

mavgtbtc (Merchant) Average time interval between batch transactions

mavgtaut (Merchant) Average time between authentications for a merchant

mratks (Merchant) Ratio of keyed versus swiped transactions

mnidclac (Merchant) Number of identical customer accounts

mnidcham (Merchant) Number of identical charge amounts

mtrzsrc (Merchant) Information source for transaction (ATM, merchant, etc.)

mtrxtrsp (Merchant) Method for sending transaction to the source (terminal, non-terminal, voice authentication)

mfloor (Merchant) Floor limit

mcbgbks (Merchant) Received charge-backs

mrtrvs (Merchant) Received retrievals (per SIC, per merchant etc.), the issuer pays for a retrieval

macqrat (Merchant) Acquisition of risk management rate (in Europe one merchant can perform multiple acquisitions but no records are kept regarding how many acquisitions were made or who performed the acquistion.)

mprevrsk (Merchant) Has a previous risk management rate occurred with this merchant? Yes or No

mtyprsk (Merchant) Type of previous risk management

(counterfeit, multiple imprint, lost/stolen/not received)

msicrat (Merchant) SIC risk management rate

mpctaut (Merchant) Percent of transactions authenticated

Network Training: When the pre-processing is complete, the fraud-related variables are fed to the network to effect the training of the network. The preferred embodiment employs a modeling technique known as a "feed forward" neural network. This type of network employs a training method to estimate parameters that define relationships between variables. While other well-known neural network training techniques may be used, the preferred training method referred to as "backpropagation gradient descent optimization" is well known to those skilled in the art.

An inherent problem of neural networks constructed using a conventional backpropagation method is a lack of generalizability. Generalizability is a measure of the predictive value of a neural network. Testing aimed at maximizing generalizablity is interpreted as involving selection of a network model with sufficient complexity so as not to underfit the data but not too much complexity so as to overfit the data. One measure of the complexity of a network is the number of hidden processing elements, and efforts to maximize generalizability have been directed to the selection from among models of different number of hidden processing elements. Unfortunately, without introducing excess complexity, in most instances it is impossible to produce all the nonlinearity necessitated by a problem by adding hidden processing elements. Most weights associated with the addition of each new hidden processing element may neither be necessary or even helpful for the modeling tasks of the near future. These excess weights tend to cause the network to conform to the characteristics or the "noise" of the

data and, accordingly, they inhibit generalization to new cases. This problem is known as overfitting and normally occurs because of excess weights.

Weight decay is a method of developing a neural network in which overfitting is minimized without sacrificing the predictive power of a model. This method initially provides a network with all the nonlinearity it needs by providing a large number of hidden processing elements. The method then involves the weights being decayed to all extents of change so that only the weights necessary for an approximation task remain. The two main premises on which the method is based are as follows: 1) When two models of equivalent performance are assigned to a training data set, the smaller model is favored; and 2) a cost function that penalizes complexity is used as part of a backpropagation algorithm. The network is trained by minimizing this cost function. Complexity is justified only as a means for expressing information contained in the data. A weight set that expresses all or almost all of the information in the data and none of the noise ensures maximum generalizability and performance.

The cost function is constructed by introducing a "decay term" to the normal error function used to train the network. The cost function is designed to optimize the model to ensure the network captures all important information in the training set without adaptation to the noise or random characteristics of the training set. With these requirements in mind, the cost function must consider not only prediction error but also the importance of the model weights. A combination of these two terms generates an object-type function that, when minimized, is optimally generalized. Executing a conventional gradient descent with this object-type function optimizes the model.

The introduction of the decay term involves an assumption about what constitutes information. The object is to select a decay term that accurately

hypothesizes the prior distribution of the weights. Finding a satisfactory prior distribution involves an examination of the possibility that the weights will possess a given distribution in the absence of data knowledge.

Weigend et al., in "Generalization by Weight-Elimination with Application to Forecasting" of Advances in Neural Information Processing Systems 3. pp. 875-82 incorporated herein by reference, disclose the following cost function for weight decay:

$$\frac{1}{2}\sum_{k\in D}(target_k - output_k)^2 + \lambda\sum_{i\in W}\frac{w_i^2 / w_o^2}{1 + w_i^2 / w_o^2} \quad \text{(Equation 1)}$$

Here:

D expresses the data set;

$target_k$ expresses the target or desired value of an element k of the data set;

$output_k$ expresses the network output for the element k of the data set;

l expresses the relative importance of the complexity term;

w expresses the weight set;

w_i expresses the value of a weight i; and

w_o expresses a constant that controls a curve shape for penalizing of weights.

The first term of the Weigend function measures the network performance, and the second term measures the complexity of the network as a term of size. While with this cost function small weights decay rapidly, large weights decay slowly or not at all.

A major drawback of the Weigend cost function and similar weight decay techniques is that they do not

accurately replicate the intended prior distribution. Finding a satisfactory prior distribution (or "prior") is a key element in creating an effective model. Most of the "priors" in the literature are effective in terms of demonstrating the concept of weight decay but lack the capacity to deal with a wide range of problems. This is because the "priors" decay weights evenly for a given processing element without effectively distinguishing important weights (those containing more information) from unimportant weights (those containing less information). This frequently results in either 1) unwanted decaying of important weights and decrease in the power of the system to accommodate nonlinearity or 2) unwanted retention of excess unimportant weights leading to overfitting.

The present invention uses the following improved cost function to deal with these problems:

$$\frac{1}{2}\sum_{k\in D}(target_k - output_k)^2 + g\lambda\sum_{i\in W}\left(c_i w_i^2 - \frac{1}{1+|w_i|}\right) \quad \text{(Equation 2)}$$

Here, g expresses a new term pertaining to the decay rate known as an interlayer gain multiplier, and C_i expresses a constant. The interlayer gain multiplier considers a relative approximation of the weights to input and output ends of the network. Accordingly, the interlayer gain multiplier affords application of a decay term with greater latent potential to elements that are closer to the inputs, the majority of the weights that are typically present in this case being more important because, while avoiding excessive decay on weights corresponding to elements closer to the outputs, their removal can effectively separate a large number of input-side weights.

By strengthening decay of the input-side weight, the cost function of Equation 2 improves the capacity of a model development constituent element 801 to decay individual weights while preserving processing elements

containing valuable information. As a result, weak interactions are eliminated and effective interactions are retained. Because, by retaining as many processing elements as possible, unnecessary individual weights are removed without loss of capacity for the nonlinearities to be modeled, the ever-present problem of overfitting is reduced.

When the cost function is iteratively applied to the network, weights decayed to a very small number (defined by "e") are removed from the network. This step, which is known as "thresholding the net", is executed because decaying weights to 0 is frequently difficult.

When the network has been trained employing past data, the definition of the network model is stored in data files. A portion of this definition referred to as the "CFG" file specifies the parameters related to the input variables of the network including such information as, for example, the length of the variables, their types, and their ranges. FIG. 21 shows a portion of a typical CFG file that specifies parameters related to an ACCOUNT variable 2101 (expressing a customer account number) and a PAUDYMDY variable 2102 (a profile variable expressing the ratio of transaction days divided by the number of days in the month).

The file formats used to store the other model definition files for the network are shown below.

ASCII File Formats

ASCII network data files (.cta, .sta, .lca, .wta) comprises tokens (non-white space) separated by white space (space, tab, new line). White space is ignored except to separate tokens. Use of line breaks and tabs is encouraged from the viewpoint of clarity but is otherwise irrelevant.

File format notation is as follows:

* Text in parentheses denotes a token.

* Text not in parentheses denotes a literal token containing characters for which accurate association is required.

* Comments on the right do not constitute part of the file format and simply provide an additional description of the format.

* The vertical lines in the comments denote blocks that can be repeated. Nested vertical lines indicate repeatable sub-blocks.

. cta Format

File format	Comments
cts	
<NetName>	
<Value>	¦ Repeated as required

cts and <NetName> must appear first. <NetName> is a standard abbreviation given in lower case (e.g., mbpn). <Value> is expressed in a sequence defined within the constant structure. If the value of a constant value is an array or structured type, the elements or fields must be separate tokens appearing in the proper sequence.

Example	Comments
cts	
mbpn	
2	InputSize
1	OutputSize
1	cHidSlabs

```
2                   HiddenSize[0]
0                   HiddenSize[1]
0                   HiddenSize[2]
3                   RandomSeed
1.0                 InitWeightMax
0                   WtsUpdateFlag
0                   ConnectInputs
0                   FnClass
1.0                 Parm1
1.0                 Parm2
-1.0                Parm3
0.0                 Parm4
0.0                 Parm5
1                   cEntTbl
0.0                 xLow
0.1                 xHigh
0.2                 HiddenAlpha[0]
0.0                 HiddenAlpha[1]
0.0                 HiddenAlpha[2]
0.1                 OutputAlpha
0.9                 HiddenBeta[0]
0.0                 HiddenBeta[1]
0.0                 HiddenBeta[2]
0.9                 OutputBeta
0.0                 Tolerance
```

```
0                   WtsUpdateFlag

0                   BatchSize

0                   LinearOutput

0                   ActTblFlag

1                   StatsFlag

1                   LearnFlag
```

In this example, HiddenSize, HiddenAlpha and HiddenBeta are all arrays and, accordingly, the elements (0, 1, 2) possess separate tokens in the sequence they appear in the type thereof.

. sta Format

File format	Comments
sts	
<NetName>	
<cSlab>	
<nSlab>	¦Repeated cSlab times
<cPe>	¦
<state>	¦¦ Repeated cPe times

sts and <NetName> must appear first. <NetName> is a standard abbreviation and is given in lower case. <cSlab> constitues the number of slabs possessing states stored in the file. The remainder of the file consists of cSlab blocks each of which describes the states of one slab. The sequence of the slab blocks in the file is not important. <nSlab> constitutes the slab number as defined in the xxx.h file. cPe constitutes the number of states for the slab. <state> constitutes the value of a single state. If the state type is an array or structured type, the elements or fields must be a separate token appearing in the proper sequence. There must be cPe <state> values in a slab block.

Example	Comments
sts	
mbpn	
6	cSlab
0	nSlab - SlabInMbpn
2	cPeIn
0.0	StsIn[0]
0.0	StsIn[1]
1	nSlab - SlabTrnMbpn
1	cPeTrn
0.0	StsTrn[0]
2	nSlab - SlabHidOMbpn
2	cPeHidO
0.0	StsHidO[0]
0.0	StsHidO[1]
5	nSlab - SlabOutMbpn
1	cPeOut
0.0	StsOut[0]
6	nSlab - SlabBiasMbpn
1	cPeBias
1.0	StsBias[0]
7	nSlab - SlabStatMbpn
3	cPeStat
0.0	StsStat[0]
0.0	StsStat[1]

.lca Format

File format	Comments
lcl	
<NetName>	
<cSlab>	
<nSlab>	¦Repeated cSlab times
<cPe>	¦
<local>	¦¦Repeated cPe times

The .lca format is the same as the .sta format except that sts is replaced by lcl. lcl and <NetName> must appear first. <NetName> is a standard abbreviation given in lowercase. <cSlab> constitutes the number of slabs that have local data stored in the file. The remainder of the file consists of cSlab blocks, each of which describes the local data value of one slab. <nSlab> constitutes the slab number as defined in the xxx.h file. The sequence of the slab blocks in the file is not important. cPe constitutes the number of local data values for a slab. <local> constitutes the value of a single local data element. If the local data type is an array or structured type, the elements or the fields must be a separate token appearing in the proper sequence. There must be be a cPe number of <local> values in a slab block.

Example	Comments
lcl	
mbpn	
3	cSlab
2	nSlab - SlabMidOMbpn

```
2                   cPe
0.0                 LclHidO[0].Error
0.0                 LclHidO[0].NetInp
0.0                 LclHidO[1].Error
0.0                 LclHidO[1].NetInp
5                   nSlab - SlabOutMbpn
1                   cPe
0.0                 LclOut[0].Error
0.0                 LclOut[0].NetInp
7                   nSlab - SlabStatMbpn
3                   cPe
0                   LclStat[0].cIter
0.0                 LclStat[0].Sum
0                   LclStat[1].cIter
0.0                 LclStat[1].Sum
0                   LclStat[2].cIter
0.0                 LclStat[2].Sum
```

In this example, the <local> values are all structured types and, accordingly, each field (Error and NetInp; cIter and Sum) possess a separate token in the sequence they appear in the type thereof.

.wta Format

File format	Comments
wts	
<NetName>	
<cClass>	

```
<nSlab>                    | Repeated cClass times
<nClass>                   |
<cIcn>                     |
<weight>                   | | Repeated cIcn times
```

wts and <NetName> must appear first. <NetName> is a standard abbreviation and is given in lowercase. <cClass> constitutes the number of the slab/class combinations that have weights stored in the file. The remainder of the file comprises cClass blocks which each describe the weights of one slab. The order of the class blocks in the file is not important. <nSlab> constitutes the slab number, as defined in the xxx.h file. <nClass> constitutes the class number as defined in the xxx.h file. <weight> constitutes the value of a single weight. If the weight type is an array or structured type the element or the fields must be a separate token appearing in the proper sequence. There should be cIcn number of <weight> values in a slab block.

Example	Comments
wts	
mbpn	
2	cClass
2	nSlab - SlabHidOMbpn
0	nClass - PeHidOMbpnFromFrev
6	cIcn
0.0	WtsHidO[PE_0][0]
0.0	WtsHidO[PE_0][1]
0.0	WtsHidO[PE_0][2]
0.0	WtsHidO[PE_1][0]

```
0.0                 WtsHidO[PE_1][1]

0.0                 WtsHidO[PE_1][2]

5                   nSlab = SlabOutMbpn

0                   nClass = PeOutMbpnFromPrev

3                   cIcn

0.0                 WtsOut[PE_0][0]

0.0                 WtsOut[PE_0][1]

0.0                 WtsOut[PE_0][2]
```

While weight values for a slab and a class are stored as a one-dimensional array, they are conceptually indexed by two values, that is to say, by PE and interconnection within PE. As illustrated here, these values are stored in a line-priority sequence.

Transaction Processing Constituent Element 802

Once the model has been created, trained and stored, fraud detection is initiated. A transaction processing constituent element 802 of system 100 preferably runs within the context of a conventional authentication or posting system for customer transactions. The transaction processing constituent element 802 reads current transaction data and customer data from databases 805, 806 and generates as output a fraud score representing the likelihood of fraud for each transaction. Furthermore, the transaction processing constituent element 802 compares the likelihood of fraud with a predetermined threshold, and is able to flag transactions for which the threshold is exceeded.

The current transaction data from the database 805 normally includes information such as transaction dollar amount, date, time (and time zone if necessary); approve/decline code, cash/merchandise code, available credit (or balance), credit line, merchant category

code, merchant Zip code, and PIN verification (if applicable).

The customer data from the database 806 normally includes information from the following three sources: 1) general information about the customer; 2) data on all approved or declined transactions in the previous seven days; and 3) a profile record containing data describing the transaction pattern of the customer over the last six months. The general information on the customer normally includes information such as the customer Zip code, the account open date, and the expiry date. The profile record is a single record in a profile database in which the transaction pattern of a customer is summarized using a moving average method. As described below, the profile record is periodically updated (normally monthly) employing all transactions for the customer for the period in question.

The system 100 can be actuated as either a batch, semi-real-time, or real-time system. The structure and processing flow of each of these variations will be hereinafter described.

Batch System: FIG. 14 shows the actuation of a batch system. Transactions are recorded throughout the day or other suitable period (1402). At the end of each day, the system executes steps 1403 to 1409 for each transaction. The system obtains data that describes a current transaction (1403), as well as past transaction data, customer data and profile data (1404). The system then applies this data to the neural network (1405) and produces a fraud score (1406). If the fraud score exceeds a threshold (1407), the account is flagged (1408). Accordingly, using the batch system, the transaction to which the highest fraud score has been assigned cannot itself be blocked but instead the account is flagged (1404) at the end of the day to ensure no future transactions are possible. While the batch system does not facilitate immediate detection of a fraudulent transaction, the use of the batch system

is unavoidable in some implementations in order to ensure response time.

Semi-Real-Time System: While a semi-real-time system is actuated by a method identical to the batch system and uses the same data files, this system guarantees that just one high score transaction is authenticated before the account is flagged. In this system, as shown in FIG. 15, a process to determine fraud likelihood is executed (Steps 1504 to 1509) immediately after a transaction is authenticated (1503). Steps 1504 to 1509 correspond to Steps 1403 to 1409 of the batch system shown in FIG. 14. If the likelihood of fraud is high, the account is flagged (1509) to prevent any future transactions. Accordingly, while similarly to a batch system a current transaction cannot be blocked, the semi-real-time system facilitates the blocking of subsequent transactions.

Real-Time System: A real-time system executes a process to determine fraud likelihood prior to a transaction being authenticated. Because of the limitations required to ensure response time, using a real-time system it is preferable that the number of database access calls be minimized. Accordingly, in this embodiment, all customer information including general information and past transaction data is found in a single record of the profile database 806. The profile database 806 is generated from past transaction and customer data prior to initiation of the transaction processing constituent element and is updated after each transaction as described below. Because all the required data is located in one place, data retrieval is quicker in this system than in the batch or semi-real-time schemes. In order to keep the profile database 806 current, profile records are updated after each transaction using moving averages where applicable.

FIG. 16 shows a flowchart of a real-time system in which a profile database is employed. When a request

for authentication of a transaction is received from a merchant (1602), the system produces profile data in which the transaction patterns of the customer are summarized (1604) and produces data about the current transaction (1603). The system then applies this data to the stored neural network model (1605). A fraud score (expressing the likelihood of fraud for the transaction) is produced and compared to a threshold (1607). Steps 1601 to 1607 are executed prior to a transaction being authenticated and, accordingly, the fraud score can be sent to an authentication system (1608) and the transaction blocked by the authentication system if the threshold has been exceeded. If the threshold is not exceeded, the low fraud score is sent to the authentication system (1609). The system then updates the customer profile database 806 with the new transaction data (1610). Accordingly, in this system, the profile database 806 is continually updated (unlike the batch and semi-real-time systems, in which profile database 806 is updated only periodically).

FIG. 12 shows a method for creating a profile record. The system executes the steps of this method when there is no existing profile record for the customer. The system reads a past transaction database 1101 for the past six months and a customer database 1103 (Steps 1202 and 1203 respectively). The system generates a new profile record (1204) of the data produced and saves this in a profile database (1205). If there remain accounts to be processed (1206), the steps 1202 to 1205 are repeated.

FIG. 13 shows a method for updating an existing profile record. The system reads the past transaction database 1101 for the past six months, the customer database 1103, and a profile database (Steps 1302, 1303, and 1304 respectively). The system combines the data into a single value for each variable in the profile database. This value is generated employing one

or two formulae.

Equation 3 is employed for variables that express an average value over a period of time (for example, average dollars of transactions in a month):

newProfData = ((1-a)*oldProfData) + (a *currentVal))

(Equation 3)

Equation 4 is employed for variables that express maximum values over a period of time (for example, highest monthly balance):

newProfData = max (currentVal, b * oldProfData)

(Equation 4)

In Equations 3 and 4:

newProfData denotes a new value for a profile variable;

oldProfData denotes an old value of the (profile) variable;

currentVal denotes the most recent value of the variable from a past transaction database; and

a and b denote decay factors used to give more importance to recent months and less importance to past months.

The value of b is set so that older data decay at a permissible rate. A typical value of b is 0.95.

The value of a is generated in the following way. For the batch and semi-real-time systems, a is set to a value at which the effect of the value of the period more than six months previous approaches zero. For profiles that have been in existence for at least six months, the value of a is set at 1/6. For newer profiles, the value of a is set to 1/(n+1) where n is the number of months since the profile was created. In the real-time system profile updates do not occur at regular intervals. Accordingly, a is determined employing the following equation:

$$a = 1-\exp(-t/T) \qquad \text{(Equation 5)}$$

Here, in this equation:

t denotes the time between the current transaction and the last transaction; and

T denotes a time constant for the specific variable.

Furthermore, the currentVal of the real-time system represents the value of a variable estimated employing only information related to the current transaction and the time since the last transaction without reference to any other time-series information.

When the new values for the profile variables have been generated they are placed in an updated profile record (1305) and saved in the profile database (1306). If there remain accounts to be processed (1307), the system repeats Steps 1302 to 1306.

In all of these embodiments a pre-processing is performed on the current transaction data and the customer data to derive fraud-related variables experientially determined to be effective predictors of fraud. This is performed employing the same technique and the same fraud-related variables associated with the neural network training as described above.

FIGS. 17 to 19 are flowcharts showing the actuation of a preferred embodiment of a transaction processing constituent element. Some of the individual elements of the flowchart are denoted by symbols that correspond to module names.

FIG. 17 shows the overall actuation of the transaction processing constituent element 802. First, the system executes a module CINITNET 1702 and this initializes the network structure. Next, the system executes a module CSCORE 1703. The module CSCORE 1703 uses current transaction activity data, data that

describes transactions over the past seven days, a profile record and customer data to generate reason codes (described below) and a fraud score that indicates the likelihood that the current transaction is fraudulent. The system then checks to see whether there are more transactions to be processed (1704) and repeats the module CSCORE 1703 for all remaining transactions. When there are no more transactions to be processed, the system executes a module FREENET 1705 that releases the network structures and allows the network structures to be used for further processing.

FIG. 18 shows the actuation of a module CSCORE 1703. First, the module CSCORE 1703 produces current transaction data, data describing transactions of the past seven days, a profile record and customer data (Steps 1802 to 1805). From these data, the module CSCORE 1703 generates the fraud-related variables (1806) described above. The system then executes a module DeployNet 1807, applies the fraud-related variables to the stored neural network, and provides a fraud score and reason codes.

FIG. 19 shows the actuation of the module DeployNet 1807. The module DeployNet 1807 initially scales the fraud-related variables 1902 to match the previous standardization executed in the model development. If the value of a variable is missing, the DeployNet sets the average value found in the training set to this value. Next, in Step 1903, the DeployNet applies the scaled variables to an input layer of the neural network 108. In Step 1904, the DeployNet processes the applied data by way of the network to generate the fraud score. The method for iterating the network is well known to those skilled in the art.

In Step 1904, in addition to providing fraud scores, the module DeployNet 1807 generates "reason codes" as an option. These codes denote which inputs to the model are most important in determining the fraud score for a given transaction. Any technique that

facilitates the recording of these reasons can be used. In a preferred embodiment, the technique described in the pending U.S. Patent Application No. 07/814,179 "Neural Network Having Expert System Functionality" by Curt A. Levey filed December 30, 1991 is used, the disclosed technique of which is incorporated herein by reference.

The following module descriptions summarize the functions executed by the individual modules.

FALCON C Files

File name: CINITNET

Description: Contains code to allocate and initialize the network structures.

Function name: CINITNET()

Description: Allocates and initializes the network structures.

File name: CSCORE

Description: Generates fraud related variables and iterates the neural network.

Function name: SCORE()

Description: Generates fraud related variables from non-processed variables and makes calls to initialize the input layer and iterate the neural network.

Function name: setInput()

Description: Sets the input value for a processing element in the input layer.

Function name: hiReason()

Description: Finds the three maximum reasons for the score.

File name: CFREENET

Description: Makes function calls to release the network structures.

Function name: CFREENET()

Description: Releases the network structures.

File name: CCREATEP

Description: Contains the cardholder profile creation code.

Function name: createpf()

Description: Creates a profile record for a cardholder using the previous month's authentications and cardholder data.

File name: CUPDATEP

Description: Updates a profile of individual valid cardholders.

Function name: updatepf()

Description: Updates a profile record for a cardholder using the previous month's authentications and cardholder data as well as previous profile record values.

File name: CCOMMON

Description: This file contains functions required by at least two of createpf(), updatepf() and score().

Function name: accumMiscCnts()

Description: Increments various types of counters for each authentication found.

Function name: accumSicCnts()

Description: Increments SIC variable counters.

Function name: initSicCounts()

Description: Initializes the SIC variable counters.

Function name: updateSicMovAvgs()

Description: Updates the SIC profile variables.

Function name: writeMiscToProfile()

Description: Writes various variables in a profile record after they have been calculated.

Function name: bncDate()

Description: Converts a Julian date to a date indicating the number of days since January 1st, 1990.

Function name: missStr()

Description: Checks for a "non-matching" flag (period) in a string terminated by a blank character (null). A string must possess only blanks and periods by which it is regarded as non-matching. A string possessing only blanks will also be regarded as "non-matching".

Cascade Operation

One method for improving system performance involves employment of a "cascade" operation. In a cascaded operation more than one neural network model is used. The second neural network model is trained by a model development constituent element 801 in the same way as described above. However, in the training of the second model the model development constituent element

801 uses only those transactions that have fraud scores above a predetermined cascade threshold as determined by prior application to the first neural network mode. Accordingly, the second model provides scores for high scored transactions of improved accuracy. While the two models can be trained using the same fraud-related variables, different variables are frequently found in the two models.

FIG. 20 shows a flowchart of the actuation of a transaction processing constituent element in a cascaded system. First, the transaction processing constituent element 802 scores each transaction using the first model (2002) as described above. The transactions of a score in excess of the cascade threshold (2003) are applied to a second neural network model (2005). The system outputs scores and reason codes from either the first model (2004) or the second model (2006) as appropriate.

The above-described cascading technique may be expanded to include three or more neural network models of which each has an associated cascade threshold.

Performance Monitor

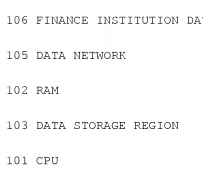
The system periodically monitors its performance based on measurement of performance metric including the fraud detection rate and the false positive rate. Other factors and statistics may be incorporated into the performance metric. When the performance metric falls below a predetermined performance level, the system is able to either inform the user that redevelopment of the fraud model is required or it can automatically implement the model redevelopment.

It is apparent from the above description that the invention disclosed herein provides a novel and effective method for detecting fraudulent use of customer accounts and account numbers, and that high detection rates thereof are achieved while keeping

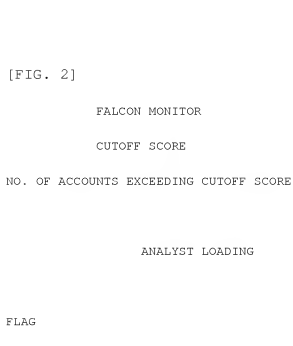
false positive rates relatively low. The foregoing discussion discloses methods and embodiments of the present invention that serve merely as examples thereof. As will be understood by those skilled in the art, the invention may be embodied in many other specific forms without departing from the gist or the essential characteristics thereof. For example, other predictive modeling techniques besides neural networks may be used. Furthermore, other variables may be used for both the model development constituent elements and transaction processing constituent elements.

Accordingly, the disclosure of the present invention is intended to be illustrative of the preferred embodiments and is not intended to limit the scope of the invention. The scope of the invention is to be limited only by the claims.

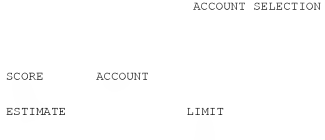
[FIG. 1]


106 FINANCE INSTITUTION DATA FACILITY
105 DATA NETWORK
102 RAM
103 DATA STORAGE REGION
101 CPU
107 PROGRAM STORAGE REGION
104 OUTPUT DEVICE

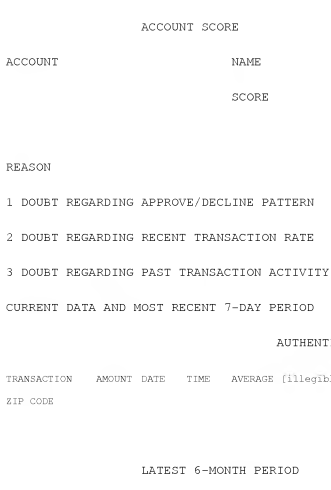
[FIG. 2]


FALCON MONITOR
CUTOFF SCORE
NO. OF ACCOUNTS EXCEEDING CUTOFF SCORE
ANALYST LOADING
FLAG
TIME
FRAUD SCORE
OK
HELP

[FIG. 3]


ACCOUNT SELECTION
SCORE
ACCOUNT
ESTIMATE
LIMIT
OK
HELP

[FIG. 4]


ACCOUNT SCORE
ACCOUNT
NAME
SCORE
REASON
1 DOUBT REGARDING APPROVE/DECLINE PATTERN
2 DOUBT REGARDING RECENT TRANSACTION RATE
3 DOUBT REGARDING PAST TRANSACTION ACTIVITY
CURRENT DATA AND MOST RECENT 7-DAY PERIOD
AUTHENTICATION RECORD
TRANSACTION
AMOUNT
DATE
TIME
AVERAGE [illegible]
CREDIT LIMIT
Sic
MERCHANT
ZIP CODE
LATEST 6-MONTH PERIOD
OK
HELP

[FIG. 5]

CARDHOLDER INFORMATION

ACCOUNT

NAME 1

NAME 2

PREFERRED CONTACT TIME

TELEPHONE NO.

DIAL HOME

DIAL WORK 1

DIAL WORK 2

ADDRESS

ADDRESS 1

ADDRESS 2

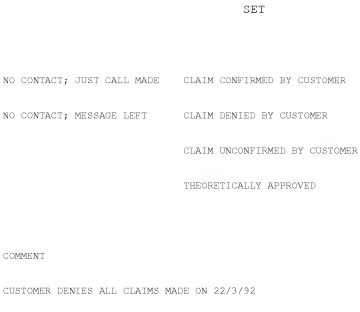
CITY

STATE ZIP CODE

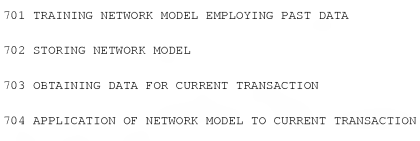
SET

OK HELP

[FIG. 6]


SET
NO CONTACT; JUST CALL MADE
NO CONTACT; MESSAGE LEFT
CLAIM CONFIRMED BY CUSTOMER
CLAIM DENIED BY CUSTOMER
CLAIM UNCONFIRMED BY CUSTOMER
THEORETICALLY APPROVED
COMMENT
CUSTOMER DENIES ALL CLAIMS MADE ON 22/3/92
OK
CANCEL
HELP

[FIG. 7]


701 TRAINING NETWORK MODEL EMPLOYING PAST DATA
702 STORING NETWORK MODEL
703 OBTAINING DATA FOR CURRENT TRANSACTION
704 APPLICATION OF NETWORK MODEL TO CURRENT TRANSACTION
705 OUTPUTTING RESULT

[FIG. 8]

804 PAST DATA

801 MODEL DEVELOPMENT

802 TRANSACTION PROCESSING

FEED INPUT DATA

CONSTRUCTION OF MODEL

RESULT FROM MODEL

FRAUD SCORE AND REASON CODE

CREATION OR UPDATING OF PROFILE

108 NEURAL NETWORK

805 CURRENT TRANSACTION DATA

808 CUSTOMER DATA (CONTAINING PROFILE)

807 OUTPUT

[FIG. 9]

903 TRANSMISSION FUNCTION

[FIG. 10]

[FIG. 11]

1101 PAST TRANSACTION DATABASE

1102 FRAUD DATABASE

1103 CUSTOMER DATABASE

1111 PROFILE RECORD

1116 1-DAY ACCOUNT VARIABLE

1119 7-DAY ACCOUNT VARIABLE

1121 modln2 DATA SET

1123 SCALED modln2 DATA SET

[FIG. 12]

1201 START

1202 READING PAST TRANSACTION DATABASE

1203 READING CUSTOMER DATABASE

1204 CREATING NEW PROFILE RECORD

1205 SAVING NEW PROFILE RECORD IN PROFILE DATABASE

1206 ACCOUNTS STILL REMAINING?

1207 END

[FIG. 13]

1301 START

1302 READING PAST TRANSACTION DATABASE

1303 READING CUSTOMER DATABASE

1304 READING RECORD FROM PROFILE DATABASE

1305 CREATING UPDATED PROFILE RECORD

1306 SAVING UPDATED PROFILE RECORD IN PROFILE DATABASE

1307 ACCOUNTS STILL REMAINING?

1308 END

[FIG. 14]

1401 START

1402 STORING OF 1-DAY OF TRANSACTIONS

1403 OBTAINING CURRENT TRANSACTION DATA

1404 OBTAINING PAST TRANSACTION DATA, CUSTOMER DATA AND PROFILE DATA

1405 APPLICATION OF DATA TO NEURAL NETWORK

1406 OBTAINING FRAUD SCORE FROM NEURAL NETWORK

1407 FRAUD SCORE > THRESHOLD?

1408 ACCOUNT FLAGGED

1409 END

[FIG. 15]

1501 START

1502 MERCHANT REQUESTS AUTHENTICATION

1503 TRANSACTION AUTHENTICATED WHEN ACCOUNT STILL NOT FLAGGED

1504 OBTAINING CURRENT TRANSACTION DATA

1505 OBTAINING PAST TRANSACTION DATA, CUSTOMER DATA AND PROFILE DATA

1506 APPLICATION OF DATA TO NEURAL NETWORK

1507 OBTAINING FRAUD SCORE FROM NEURAL NETWORK

1508 FRAUD SCORE > THRESHOLD?

1509 ACCOUNT FLAGGED

1510 END

[FIG. 16]

1601 START

1602 AUTHENTICATION REQUESTED BY MERCHANT

1603 CURRENT TRANSACTION DATA OBTAINED

1604 PROFILE DATA OBTAINED

1605 DATA APPLIED TO NEURAL NETWORK

1606 FRAUD SCORE OBTAINED FROM NEURAL NETWORK

1607 FRAUD SCORE > THRESHOLD?

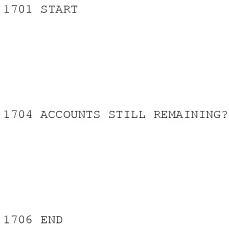
1608 SIGNAL INDICATING HIGH FRAUD SCORE SENT TO AUTHENTICATION SYSTEM AND ACCOUNT FLAGGED

1609 SIGNAL INDICATING LOW FRAUD SCORE SENT TO AUTHENTICATION SYSTEM

1610 PROFILE DATA UPDATED

1611 END

[FIG. 17]



[FIG. 18]

1801 START

1802 CURRENT TRANSACTION DATA OBTAINED

1803 OTHER TRANSACTION DATA FOR PAST 7 DAYS OBTAINED

1804 RECORD OBTAINED FROM PROFILE DATABASE

1805 CUSTOMER DATA OBTAINED

1806 FRAUD RELATED VARIABLES CREATED

1807 DeployNet EXECUTED

1808 SCORE AND REASON CODE OUTPUT

1809 END

[FIG. 19]

1901 START

1902 FRAUD-RELATED VARIABLES SCALED

1903 INPUT LAYER OF NEURAL NETWORK INITIALIZED

1904 GENERATION OF SCORES AND REASON CODES ITERATED IN NETWORK

1905 END

[FIG. 20]

2001 START

2002 TRANSACTION SCORES ASSIGNED EMPLOYING FIRST MODEL

2003 SCORE > CASCADE THRESHOLD?

2004 OUTPUT OF SCORES AND REASONS FROM FIRST MODEL

2005 TRANSACTION SCORES ASSIGNED EMPLOYING SECOND MODEL

2006 OUTPUT OF SCORES AND REASONS FROM SECOND MODEL

2007 END

[FIG. 21]

[図１]

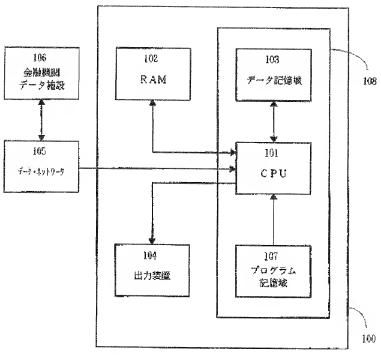

106
金融機関
データ施設
102
RAM
103
データ記憶域
108
105
ﾃﾞｰﾀ･ﾈｯﾄﾜｰｸ
101
CPU
104
出力装置
107
プログラム
記憶域
100

FIGURE 1

[図２]

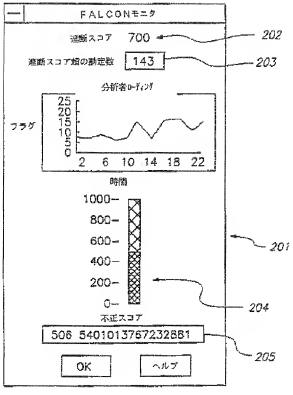

FALCONモニタ
遮断スコア 700
202
遮断スコア超の静定数 143
203
分析者ローディング
フラグ
25
20
15
10
5
0
2 6 10 14 18 22
時間
1000
800
600
400
200
0
201
204
不正スコア
508 540101367232881
205
OK
ヘルプ

FIGURE 2

[図３]

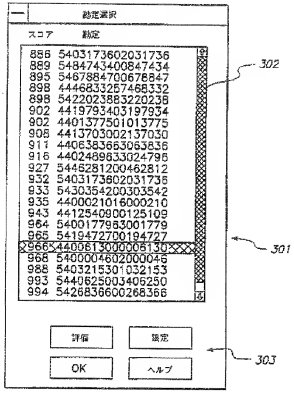

設定選択
スコア　設定
886 5403173602031736
889 5484743400847434
895 5467884700676847
898 4446833257468332
898 5422023883220236
902 4419793403197934
902 4401377501013775
906 4413703002137030
911 4406383663063836
916 4402489833024796
927 5445281200462812
932 5403173602031736
933 5430364200303542
935 4400021016000210
943 4412540900125109
964 5400177983001779
965 5419472700194727
966×4400613200206130
966 5400004602000046
988 5403215301032153
993 5440625003406250
994 5426836600268366
評価
設定
OK
ヘルプ
302
301
303

FIGURE 3

【図４】

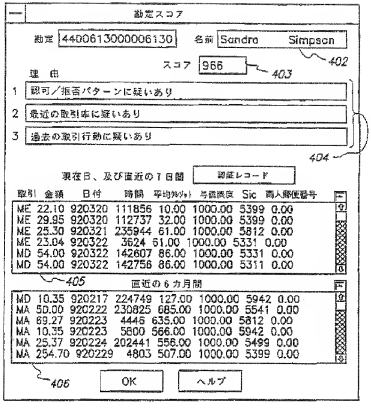

勘定スコア
勘定 4400613000006130
名前 Sandra Simpson
402
スコア 966
403
理 由
1 顧可／拒否パターンに疑いあり
2 最近の取引本に疑いあり
3 過去の取引行動に疑いあり
404
現在日、及び直近の７日間
照会レコード
取引 金額 日付 時間 平均ﾍﾙｽ 与信限度 Sic 商人郵便番号
ME 22.10 920320 111856 10.00 1000.00 5399 0.00
ME 29.85 920320 112737 32.00 1000.00 5399 0.00
ME 25.30 920321 235944 61.00 1000.00 5812 0.00
ME 23.04 920322 3624 61.00 1000.00 5331 0.00
MD 54.00 920322 142607 86.00 1000.00 5331 0.00
MD 54.00 920322 142756 86.00 1000.00 5311 0.00
405
直近の６カ月間
MD 10.35 920217 224749 127.00 1000.00 5942 0.00
MA 50.00 920222 230825 685.00 1000.00 5541 0.00
MA 69.27 920223 4446 635.00 1000.00 5812 0.00
MA 10.35 920223 5800 566.00 1000.00 5942 0.00
MA 25.37 920224 202441 558.00 1000.00 5499 0.00
MA 254.70 920229 4803 507.00 1000.00 5399 0.00
406
OK
ヘルプ
401

FIGURE 4

[図５]

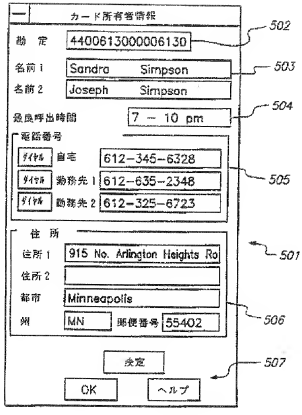

カード所有者情報
502
勘定
4400613000006130
名前１
Sandra Simpson
503
名前２
Joseph Simpson
504
優良呼出時間
7 – 10 pm
電話番号
ダイヤル
自宅
612-345-6328
ダイヤル
勤務先１
612-635-2348
505
ダイヤル
勤務先２
612-325-6723
住所
501
住所１
915 No. Arlington Heights Ro
住所２
都市
Minneapolis
州
MN
郵便番号
55402
506
決定
507
OK
ヘルプ

FIGURE 5

[図６]

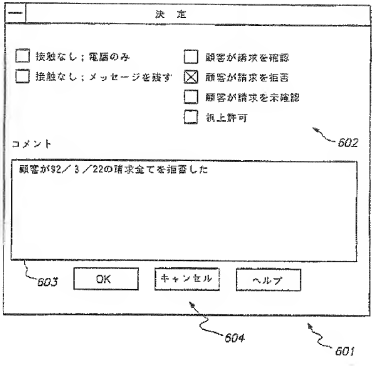

決　定
接触なし；電話のみ
接触なし；メッセージを残す
顧客が請求を確認
顧客が請求を拒否
顧客が請求を未確認
訴上許可
602
コメント
顧客が92／３／22の請求金てを拒否した
603
OK
キャンセル
ヘルプ
604
601

FIGURE 6

【図７】

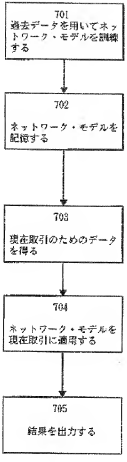

701
過去データを用いてネットワーク・モデルを訓練する
702
ネットワーク・モデルを記憶する
703
現在取引のためのデータを得る
704
ネットワーク・モデルを現在取引に適用する
705
結果を出力する

FIGURE 7

【図８】

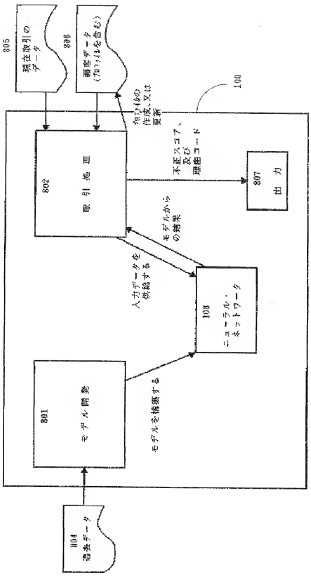

805
現在取引のデータ
808
顧客データ（プロフィルを含む）
100
分析結果の作成、又は更新
802
取引処理
不正スコア、及びコード
807
出力
モデルからの結果
入力データを供給する
108
ニューラル・ネットワーク
801
モデル照会
モデルを構築する
804
過去データ

FIGURE 8

【図９】

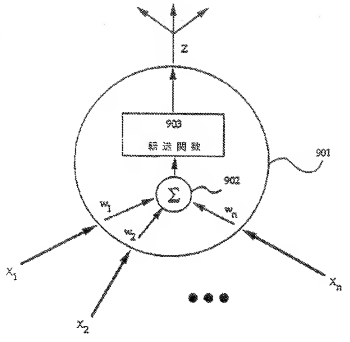


FIGURE 9

[図１０]

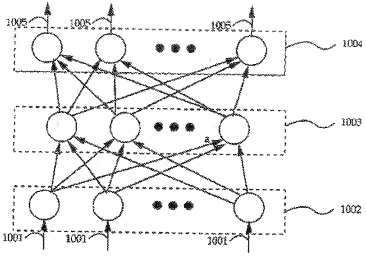


FIGURE 10

【図１１】

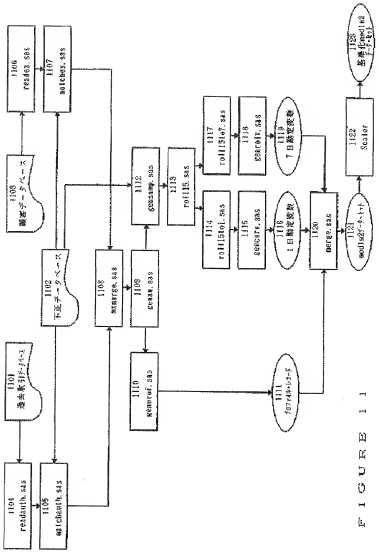


FIGURE 11

【図１２】

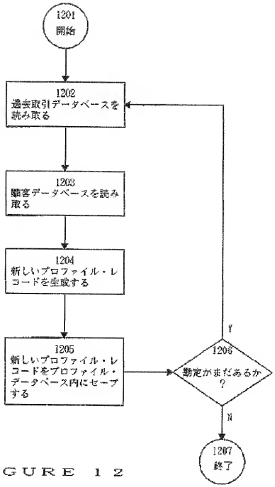


FIGURE 12

【図１３】

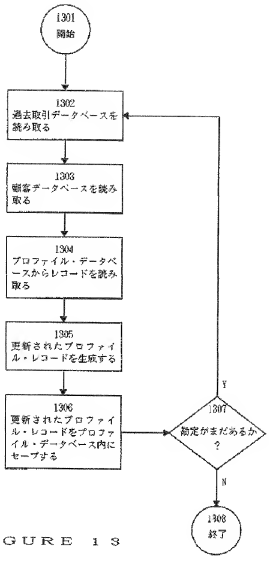


FIGURE 13

【図１４】

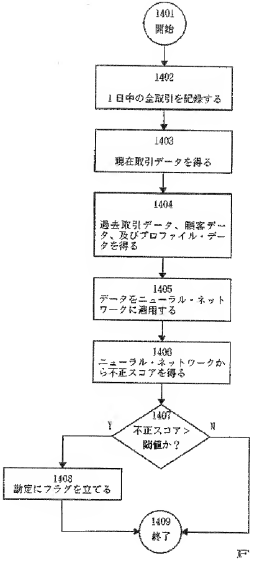


FIGURE 14

【図１５】

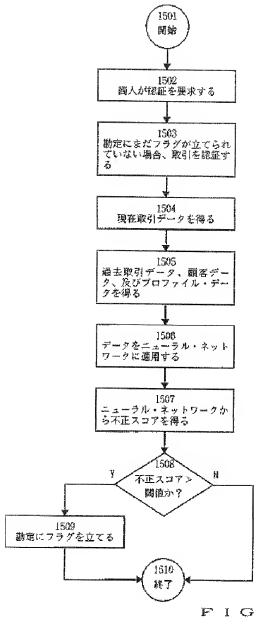


FIGURE 15

[図１６]

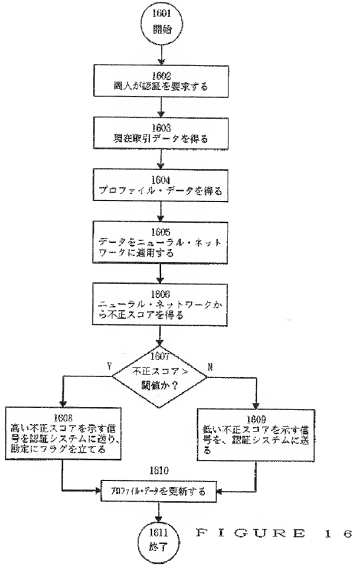


FIGURE 16

【図１７】

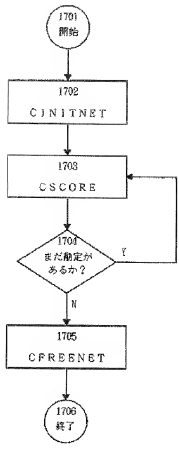


FIGURE 17

【図１８】

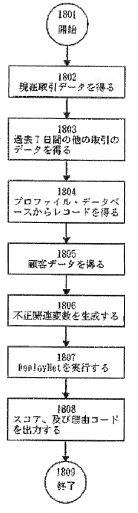


FIGURE 18

【図１９】

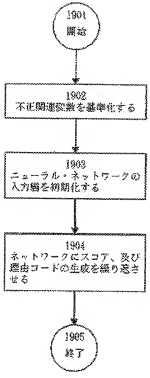


FIGURE 19

【図２０】

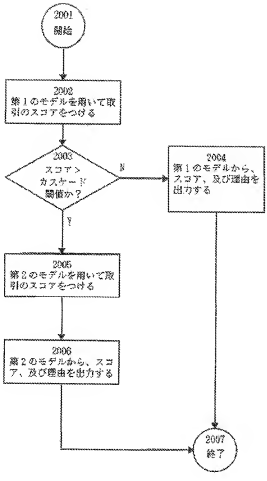


FIGURE 20

[図２１]

```
Record Length=784

0
Name= ACCOUNT
Type=EXCLUDE
Slab=NONE
Size=0
Start=0 Length=16
RecCnt=0
Min=1.7976931e+308 Max=-1.7976931e+308
MissingValue=0.
Sum=0.
Mean=0.
StdDev=0.
Derivative=0                                   2101
TimeSlice=0
NbrOfSymbols=0
Symbolic=NUMERIC
ScaleMode=AUTO ScaleFn=LIN
DivFlag=0
Divisor=0. Range=0.

0
Name=PAUDYMDY
Type=CONTINUOUS
Slab=INPUT
Size=1
Start=16 Length=12
RecCnt=23312
Min=3.22581e-002 Max=1.
MissingValue=0.18761507
Sum=4373.6825                                  2102
Mean=0.18761507
StdDev=0.13174467
Derivative=0
TimeSlice=0
NbrOfSymbols=0
Symbolic=NUMERIC
ScaleMode=AUTO ScaleFn=LIN
DivFlag=0
Divisor=0. Range=0.9677419
```

FIGURE 21